QL Today

Volume 6
Issue 1
May/June
2001

ISSN 1432-5454

The Magazine about QL, QDOS, Sinclair Computers, SMSQ...

Holidays combined with a QL Show in one of the nicest areas of Germany?

Or even "QL 2001"!

Hot News! See page 4

Page 14

Another German/Austrian QL Show is on schedule!
Highlight it now on your Agenda!

Contents

Advertisers

in alphabetical order

QL *Today*

ISSN 1432-5454

German office & Publisher:

Jochen Merz Software
Im stillen Winkel 12
47169 Duisburg
Germany

Tel. +49 203 502011
Fax +49 203 502012
Box1 +49 203 502013
Box2 +49 203 502014
Mobile +49 1701 222277
email: JMerz@J-M-S.com
email: jochen@QLToday.com

English office:

Q Branch
20 Locks Hill
Portslade
BN41 2LB
United Kingdom

Tel. +44 1273 386030
Mobile +44 7836 745501
Fax +44 1273 381577
email: qbranch@qbranch.demon.co.uk

Editor:

Dilwyn Jones
41 Bro Emrys
Tal-Y-Bont, Bangor
Gwynedd
United Kingdom LL57 3YT

Tel. +44 1248 354023
Fax +44 1248 354023
email: dilwyn.jones@dj.softnet.co.uk
email: dilwyn@QLToday.com

Co-Editor:

Bruce Nicholls
38 Derham Gardens
Upminster
Essex RM14 3HA
United Kingdom

Tel +44 1708 510764
Fax +44 870 0568755
email: qltoday@q-v-d.demon.co.uk
email: bruce@QLToday.com

Please send comments and articles to
editor@QLToday.com

QL *Today* is published bi-monthly, our volume begins on beginning of June. Subscriptions begin with the current issue at the time of sign up. Please contact the German or English office for current subscription rates.

We welcome your comments, suggestions and articles. YOU make QL *Today* possible. We are constantly changing and adjusting to meet your needs and requirements. Articles for publication should be on a 3.5" disk (DD or HD) or sent via Email or into one of the JMS-BBS's. We prefer ASCII, Quill or text87 format. Pictures may be in _SCR format, we can also handle GIF or TIF or JPG. To enhance your article you may wish to include Saved Screen dumps. PLEASE send a hardcopy of all screens to be included. Don't forget to specify where in the text you would like the screen placed.

Article and Advertising deadlines are as follows:

Issue 1: 30 April	Issue 2: 30 June
Issue 3: 30 August	Issue 4: 30 October
Issue 5: 30 December	Issue 6: 28 February

QL *Today* reserves the right to publish or not publish any material submitted. Under no circumstances will QL *Today* be held liable for any direct, indirect or consequential damage or loss arising out of the use and/or inability to use any of the material published in QL *Today*. The opinions expressed herein are those of the authors and are not necessarily those of the publisher.

Editorial

Dilwyn Jones

With this issue we enter our sixth year of publishing QL Today and I would like to express my thanks to all our readers, contributors and advertisers for their enduring enthusiasm and support of the QL system.

Who would have thought way back in the early 1980s that the QL concept would have maintained such a level of interest nearly 20 years later!

While the original QL hardware is now considered by many to be a bit long in the tooth, and microdrive tapes are now all but a distant memory for most of us, the enduring appeal of the QL's unique operating system and the ease with which the computer can be programmed with its on-board BASIC language interpreter and so on mean that it still has a wide appeal and sizeable following. Just do a search on the internet for QL-related material and you may be pleasantly surprised by how much is out there!

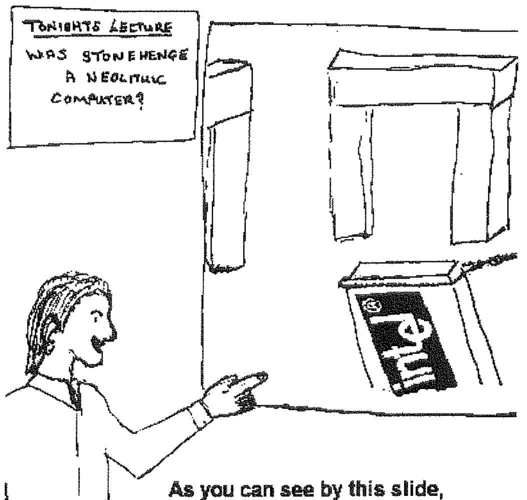
Where does the QL system go from here? Peter Graf is certain that the future of the system lies in good hardware-based native QL systems such as his Q40 and Q60 designs, while others like myself find the QL emulators route to be either an adequate or the best option for our computing needs (I see it as getting the best of both worlds).

Peter Graf, and his brother Claus, clearly do not agree with this viewpoint, as you will see from articles in this issue. I agree with what they say that QL Today should not cover too much irrelevant material relating to other computers (unless directly relevant to the QL) - after all, you subscribe to QL Today to get QL news and information - if you wanted a PC magazine you'd go to your newsagent to buy one. Some of their viewpoints are bound to stimulate debate, but their message is clear: we need to maintain a native hardware path for the future of the QL, and that they are firmly of the opinion that Peter's Q40/Q60 design path is the best one to follow. Nasta's long-awaited Goldfire project is another possible native hardware upgrade path for users of the QL, of course, once that finally becomes available.

Irrespective of your viewpoint on the emulators versus native hardware issues raised, the plain fact of the matter is that there is a good choice of emulators of the QL on various platforms, as shown by my QL Emulators CD for example, but at the moment we need to focus more on the native QL hardware. We don't need to worry about the emulators at the moment, they are taking care of themselves, we need to focus more on native QL systems which do not rely on other operating systems. The risk of emulators (according to many I have spoken to) is that apart from being limited by the underlying operating systems, they may actually encourage users to migrate to other computer systems.

What do you think? Please read the articles from the Graf brothers and let us know your opinions on the subject, whether you agree or disagree with their viewpoints, since a good debate on the issue is very important. I look forward to our letters pages bulging with debate on the issue!

Cartoon



As you can see by this slide, when we raised the altar stone we realised why the project was never finised.

NEWS

RWAP Software News

from Rich Mellor

Q-Route v2.00 has now been released which makes use of the enhanced colour drivers on QPC, QXL and the Q40. Upgrades cost 5 pounds.

It should also be noted that my email is now up and running again, address is:
RWAPsoftware2@activemail.co.uk

TURBO News

from George Gwilt

A new version 4.9 of Turbo compiler is now available from the Other Software Page on the Dilwyn Jones website. The changes in the recent version 4.8 and the newer version 4.9 are listed below.

v4.8

1. The changes in v4.8 do not introduce any new facilities. Instead they are designed to speed up and shorten the compiled programs. This is achieved by including a new section of code in Codegen which improves the output from Parser before the original code produces the compiled program.

This was envisaged from the earliest stages of Turbo, but was never put into effect until now. Version 4.8 owes its existence to detailed suggestions from Simon Goodwin.

The greatest increase in speed was obtained by recoding the increment or decrement of integer variables. Thus

x% = x% + 42

for example should now be around 70% faster. A speed up of 75% is obtainable by using the FAST option. However, in this case, there is no check that the resulting value of x% is within the limits for an integer. Thus 32740 + 42 becomes -32754 and no error is signalled. The BRIEF version does have such a check.

2. It is now possible to PRINT to the device NUL (in its SMSQ/E version) without an overflow error.

v4.9

1. Parser scans the program to be compiled and produces intermediate code which it sets in a buffer allocated from the common heap. If the buffer is too small the intermediate code is transferred to a temporary file created in the directory chosen for the compiled program. The intermediate code from which Codegen produces the compiled program is thus either in a buffer or in a file. In v4.8 there were errors in Codegen when the code was in a file. This has now been corrected.

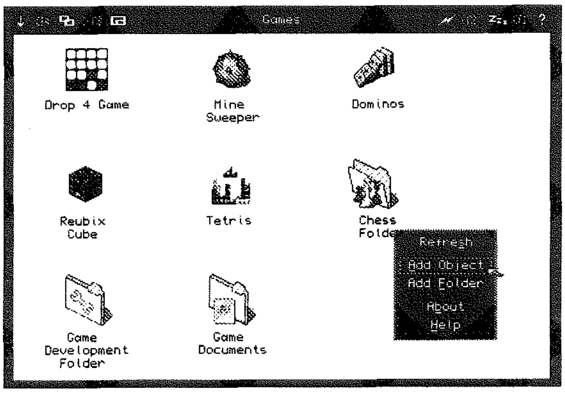
JDH Technologies

I would like to announce the on going development of a major new project for the QL family of computers, the QL DeskTop (QDT). QDT is a modern day desktop environment, similar to what is found in other operating systems such as Macintosh and Windows. For those not familiar with such desktop environments, the intent is to provide a convenient way to organize, manage, and launch programs and files.

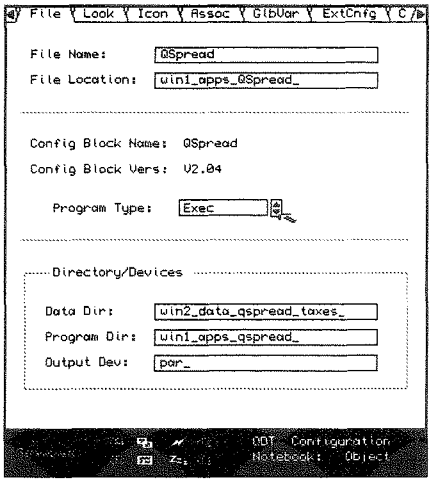
I will be giving a talk and getting feedback at the North American QL Show in June. Hopefully a first look will be published after the show.

While the project has been in development for the last two years, the time allotted to it has been very sporatic. At this point, all major obsticals have been looked at and tested, enough base code has been written, and the design architecture is complete enough to assure that the project can be completed. There is still a tremendous amount of work to do, so the availability date is not assured. My intent is to release it before the end of the year.

Jim Hunkins



Typical QDT Folder with Program and Folder Icons



File Name: QSpread
File Location: win1_apps_QSpread_

Config Block Name: QSpread
Config Block Vers: V2.04

Program Type: Exec

Directory/Devices
Data Dir: win2_data_qspread_taxes_
Program Dir: win1_apps_qspread_
Output Dev: par_

First Page of Configuration Notebook for a Program

UK General Election

Geoff Wicks

I have put a new version of my General Election analysis program on my web page.

It allows you to plot 8 new maps. These are for the 2nd, 3rd and 4th places of the parties; the areas of strength for each of the main parties; and the marginal seats.

In the seats section it is now possible to look for 2nd, 3rd and 4th place seats. I have also implemented Malcolm Cadman's suggestion of marking the marginal seats.

Sprite Editor

Jerome Grimbert has updated the sprite editor for Q40. (now version 0.99)

Details are available (in French) at

http://www.altern.org/grimbert/quidnovi.html

Also there are some more screen captures of this version (I cannot decently call it a tutorial). See

http://www.altern.org/grimbert/ql/sprted/index.html

QL BBS Situation

Derek Stewart wrote:

looks like Nene Valley BBS is closing down at the end of May this year. If anyone requires a BBS to call then my system, Holborn View is open 24 hours on a dedicated telephone line.

The system runs on a Supergold card QL, Superhermes, USR Courier modem.

Connect speeds upto 33600 baud, but due to the rubbish telephone lines 28800 and sometimes 31200 is available

There are lots of interesting QL specific software for download, which comes from all the QL Internet sites.

I have links to the Fidonet, which can give other computer interests.

If anyone is interested then please call.....

Holborn View
Tel. +44-(0)1773-74135 24 hours
Speeds 300-33600
Local and worldwide - QL user interaction - Friendly sysop - Upto date QL software to download with a QL...

QL Today has recently tried to contact Phil Borman but did not get a reply in time before this issue had to be completed to be ready for print. The next issue of QL Today will be ready by the end of July/beginning of August so you will have probably found out by then. We hope to have more news in the next issue.

Peter and Claus Graf have both supplied articles for this issue and in them they have some harsh words for some previous contributors, traders and authors.

We are sure that the people attacked in these articles will want, and, indeed, deserve, a right to reply to them but we are very aware that this could turn into a long running battle of words which many readers would eventually find boring. We have, therefore, decided to impose a strict cut-off point. The next issue will contain any replies to the articles and the following one will allow Peter and Claus space for a final comment. We hope that both parties will respect our stance and not turn this into an acrimonious and damaging public spat. We are, after all, a community of users of QDOS/SMSQ software and hardware and would be harm that community if it were to become riven by an ideological split.

Other readers are invited to comment for the next issue.

Credit Cards & Orders

Jochen Merz

Some months ago there was a discussion about foreign payment, especially for small amounts of money.

Eurocheques disappear everywhere (in fact, only very few countries are left which handle them, but even they will stop by the end of this year).

International money transfers cost a fortune, so they are no solution.

This does not leave many options, of which the best seem to be credit cards. I have never had any kind of credit card fraud during all the years I accept credit cards in the QL scene, and I think that QL customers did not worry about using their cards for mail orders.

For those still worrying: the card companies have put another obstacle in the way of credit card abuse: the card security number.

Please note that this number is obligatory for any mail order credit card purchase from October, 1st, 2001, on. This number only exists on the reverse side of your credit card. It is a 3-digit number for Visa, MasterCard, Diners, and 4 digits for Amex.

The number will not be printed on your card statement, nor will it appear on your credit card slips or receipts.

J-M-S will not put this number on file, so please specify this number with EVERY order (you do not need to give the other card details, so no connection can be made by 3rd parties!).

Quanta AGM, Portishead, 29th April 2001

A report by Dilwyn Jones

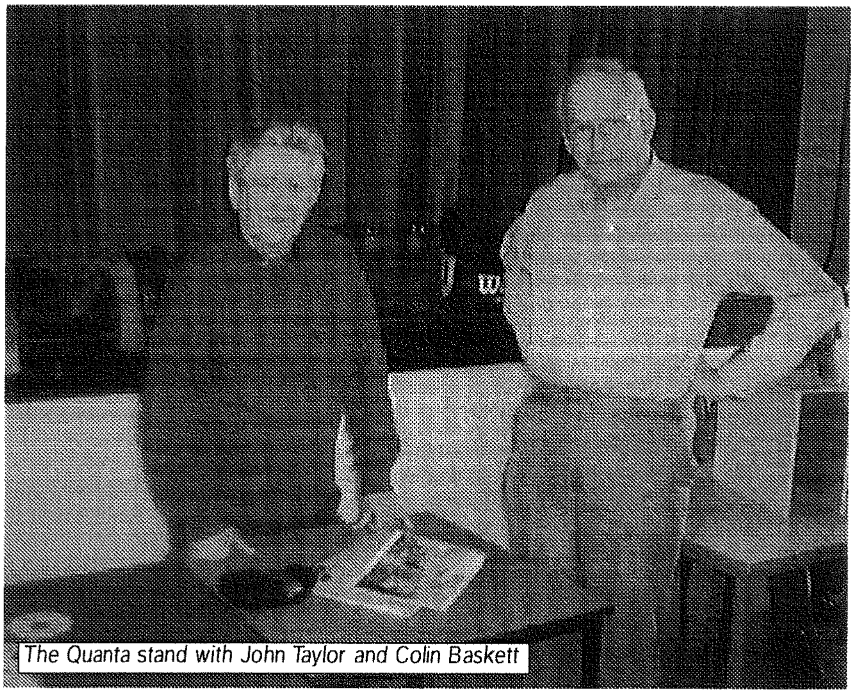
Somerset Hall in Portishead, near Bristol, was the venue for the 2001 Quanta annual general meeting. Having proved to be a popular and well attended venue in the past, this meeting proved to be somewhat of a disappointment in terms of attendance, although many present were quick to point out that the publicity for this event had not been as good as for some previous events. The local QL user group had disbanded not so long ago which may not have helped.

Nonetheless, a fairly small but enthusiastic crowd gathered towards lunchtime and early afternoon to this spacious and well liked venue. Somerset Hall is located close to the motorway and the town centre, so access was pretty easy. The hall itself is upstairs, and consisted of a meeting room at the back, a large hall well stocked with tables and chairs and power points for the assembled QL systems to entertain their owners and visitors. The hall also has a small catering facility, so hot drinks and snacks were available for those who did not wish to venture to seek refreshments elsewhere on a Sunday when many places may have been closed.

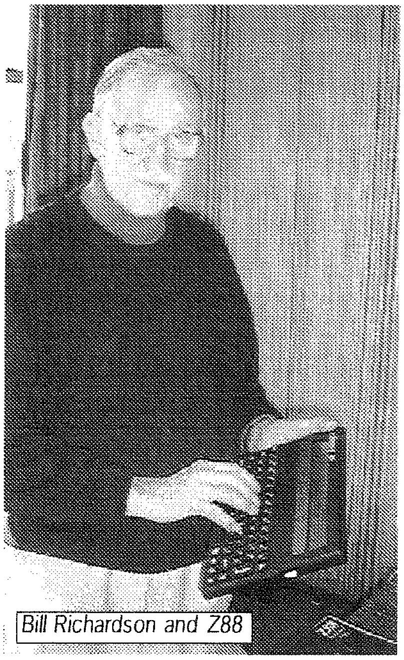
The usual gang of traders were present, including Jochen Merz Software, Geoff Wicks of Just Words, Darren Branagh of Q-Celt Computing, Quanta committee, Bill Richardson, Tony Firshman (TF Services), and QBranch along with a small number of well known QL hardware and software experts to demonstrate their wares and try to help with any problems the QLers may have.

One person who definitely needed help on the day was Roy Brereton, the Quanta secretary (and a previous chairman and head software librarian). Roy had decided before coming to the show to dismantle his tower cased QL system which turned out to be rather less than perfecty behaved on the day. True to the spirit of a QL meeting like this, there was no shortage of people willing to try to help him. The first stage in sorting the problem was to discover a loose wire in his monitor cable, but although this restored the display, the QL was in a crashed state and further work was required. At various points during the day Roy and Tony Firshman could be seen wielding a soldering iron to the poor computer!

Another person having some slight problems, this time with an Amiga computer, was Simon Goodwin. Anxious to demonstrate his digital camera link software and QL sampled sound system, Simon found himself having to work with the lid of the computer case open and occasionally having to unplug hard drives and 68040 processor cards and do some juggling of emulator files. Although it looked like hard work to me, Simon seemed to be enjoying it and getting some fantastic results transferring JPEG camera files via a serial link to Amiga QDOS, using the BASIC keywords he has written for this purpose. He said that he had also got the system working to a QL equipped with SuperHermes. The software is specific to certain Kodak cameras, but is not yet ready for general release. Using Dave Westbury's Photon software running on QDOS on his Amiga 1200, Simon demonstrated some impressive pictures of people taken at the show. He even demonstrated a picture viewed in Mode 4 - many could not believe the quality of the pictures using just the 4 colour screen mode, let alone what they would have been like in higher colour modes. Simon



The Quanta stand with John Taylor and Colin Baskett



Bill Richardson and Z88

also demonstrated an interesting custom character mapped 40x28 character display, with an animation routine based on SuperBASIC modifying the content of the character set in RAM. This was achieved by reprogramming the copper and blitter in the Amiga to produce a custom display based on using sufficiently low processor time to allow reasonable speed of operation in this mode. In a way, this animation reminded me of the processes we went through to achieve high resolution and animation on earlier computers like the ZX81, but despite this it was still an impressive display even though we know to expect some remarkable programming from Simon as it is!

Mike Ashford and other local members tried to man the Bring and Buy stall, but for once this proved to be surprisingly poorly stocked, perhaps not surprisingly in view of the poor turn out early in the day.

Tony Firshman had his range of hardware products on show, and confirmed he had no new products planned in the short term due to the pressure of work he has been under. Stocks of the existing products are still available of course.

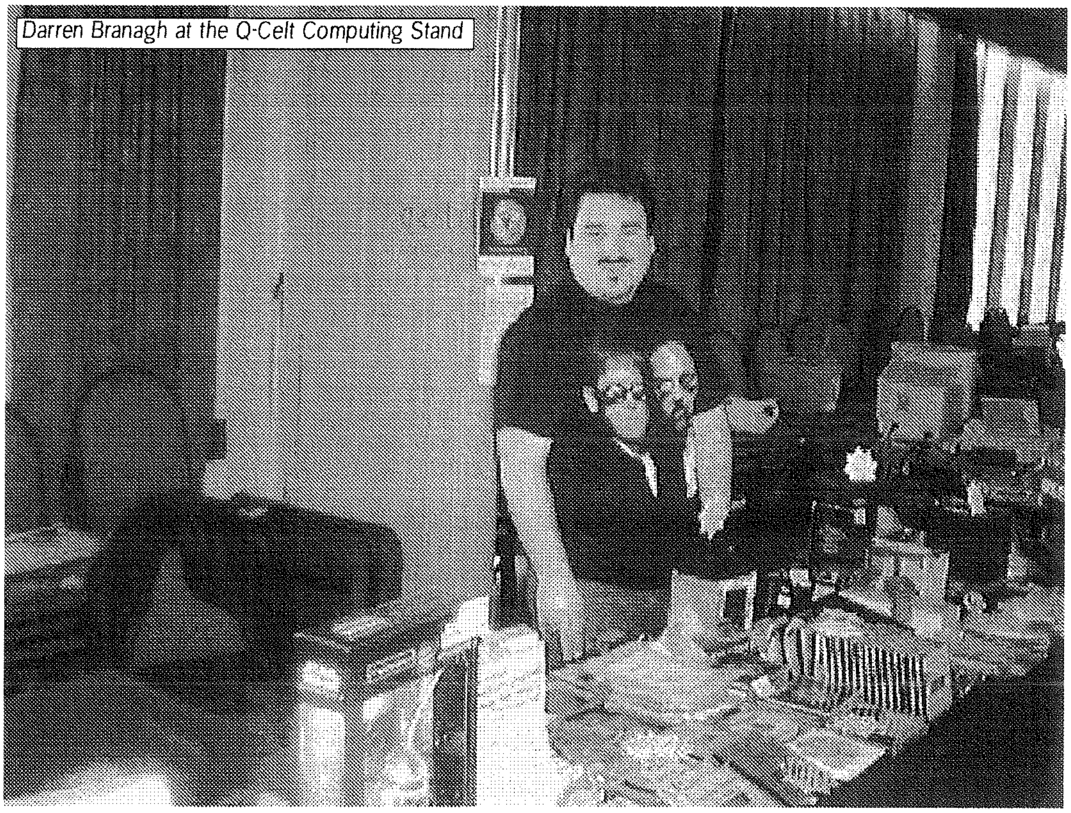
Bill Richardson had stocks of Z88 products at the show, along with a few other bits and pieces such as second hand software. Bill has a market for Spectrum hardware, in particular any Interface 1's going spare would be appreciated for his customers in Russia.

Geoff Wicks had his software products on sale, these are now neatly split into his 'tennerware' and 'poundware' lines (10 pounds and 1 pound each, obviously). His next product, currently with Beta-testers, is a Rhyming Dictionary. Geoff gave me a quick demonstration of the software, which finds words which rhyme with given words. For example, his surname (Wicks) found rhymes such as Tricks and even words ending with 'ix' such as 'fix', which is important since it demonstrates that the program is able to process words on the basis of how they sound rather than how spelled, since rhyming is a verbal issue rather than a written one. For those who can make use of such a utility, this product should be an interesting one. Geoff has also produced an Advanced Cryptics Dictionary for his Solvit Plus program. This dictionary contains over 239,000 words and phrases compiled by crossword users for crossword users. When expanded, it needs at least HD or ED disk capacity, that is, will not fit on a DD disk. This costs just 1 pound on floppy disk from Just Words.

Jochen Merz was showing a new version of Q-Spread, along with the many products he normally offers, ranging from software to back issues of that fine magazine QL Today and even some cheap CD-Rs. Jochen had suffered both from a tragic loss of his PC laptop which had been sent away for repair and returned in damaged condition and from a major car breakdown a few days before the show, so he'd had to borrow his father's car for the journey, and seemed unusually to be mourning the loss of his PC!

Back to the subject of CDRs, and the Q-Celt Computing stand currently offers just about anything you could hope for, from a range of QL software CDs to disks, novelty mice and the odd second hand Trump Cards, Gold Cards and QXLs. Quite how Darren managed to lift the huge suitcase full of goodies is beyond me, let alone how it got through customs from Ireland!

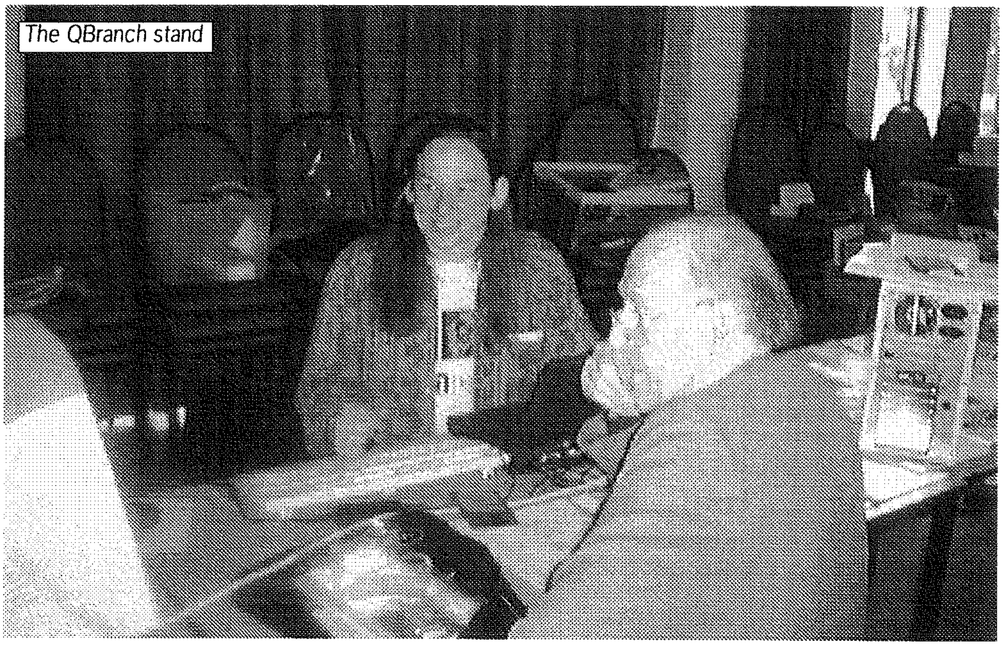
The long awaited and overdue QL Emulators CD was finally on sale. This CD includes QL emulators for several platforms and hundreds of free QL software programs (Shareware, freeware, PD programs etc) and at just 4 pounds each at the show, the CD raised a lot of interest from users of PCs, Linux, Amigas, Ataris and Apple Macs. At this price it is probably worth obtaining just to get all those QL programs! The CD itself can be freely copied in Britain and Europe - the intention behind it was not profit but rather to promote the QL and give some of th emany ex-QLers a route back into the QL scene by using emulators on whatever platform they are currently using. The CD-R cannot currently be copied or used in North America because the CD includes QL ROM images which are subject to rights held by Frank Davies and Paul Holmgren over there. Consideration is being given to producing a version of the CD including a copy of Minerva QL ROM version 1.89 which has been


Darren Branagh at the Q-Celt Computing Stand

released for use with emulators worldwide - many of the emulators need a copy of a QL ROM image to work. Q-Celt have also released a Spectrum emulation CD with a large number of programs on it, and also a CD consisting of Infocom style adventure games, the CD having been prepared by Phoebus Dokos, a Greek QLer currently living and working in the USA.

Jonathan Hudson, the author of many highly regarded QDOS programs, was sat quietly in one corner bevering away on TCP/IP for QDOS. A little bird told me he was working on porting a TCP/IP implementation from another platform to the QL in parallel to the work Jonathan Dent is doing, but that this one was having problems due to the amount of time Jonathan Hudson is having to be away from home with his work and also a mystery bug which caused the software to work some times and not at others, which by the day of the show at any rate he had not been able to locate. It is unclear if and when this software might ever see the light of day at the time of writing. Another 'little bird' was the rather cute penguin jumper his wife Daria had created for him, the penguin in question being the infamous Linux penguin symbol of course, since Jonathan is a fan on Linux and the uQLx QL emulator running of the Linux platform.

The Quanta Annual General Meeting took place at 4PM and most issues were quickly and routinely agreed and voted upon. The existing Quanta committee was returned unopposed, although Roy Brereton relinquished the mantle of Head Software Librarian to John Gilpin of the North East Manchester sub-group. John had been proposed for a committee post, but due to a procedural problem (the member who proposed him had found out his membership had expired and thus technically he was not able to propose) John Gilpin could not immediately be appointed, however the committee used powers to co-opt John Gilpin to the position of


The QBranch stand

Librarian since Roy Brereton wished to relinquish that position. Roy stays on as Quanta secretary.

The committee reported that the downward trend in membership numbers seen in recent years had slowed considerably, and they surmised that it seemed Quanta was now at a level where the membership consisted of die-hard QLers who were mostly likely to stay with Quanta and the QL. Although the previous year had shown Quanta trading slightly in the red, it was sustainable and no drastic immediate action was necessary. Colin Baskett, the newsletter editor, expressed concern at the disappointing number of contributions to the newsletter content and a discussion followed on a proposal to reduce the number of issues per year. Although a show of hands showed a majority in favour of this in order to maintain the quality and amount of content, some fears were raised that this would concern members whose main reasons for joining Quanta were to receive a regular newsletter and that reducing the number of issues might send out the wrong signals despite the good intentions. A compromise proposal was made whereby discussion is to take place on whether to reduce the number of issues per year from 12 to 10 (e.g. a 2 month issue over the Christmas and quiet summer periods) which could have the effect of saving slightly on costs without the problems inherent in going bi-monthly such as issues falling in the wrong month to publicise workshop meetings and the annual general meeting notices.

Other subjects discussed included the successful QL2000 meeting in Portsmouth, which gave QLers worldwide the opportunity to congregate under one roof for a 2 day event to celebrate the QL in the year 2000. The cost of staging a 2 day event like this may mean that Quanta would not be able to afford to run such a large event each year and in any case it would become less of a special event if held annually and may detract from Quanta's ability to hold other workshops. Proposals such as the possibility of members making additional contributions to fund the event were discussed and when the contributions reached a level sufficient to help finance the meeting, such a meeting could be planned, although the event should be tied in with some other significant event to make the meeting special.

2 Pandora QL cases from NEMQLUG

The NEMQLUG (North East Manchester) sub-group had been approached with a view to ensuring that if Quanta took the step of concentrating on two workshops in Britain each year, it would be a wise move to try to hold one in the north of England and the other in the south, possibly with the next AGM being held again in the Manchester area.

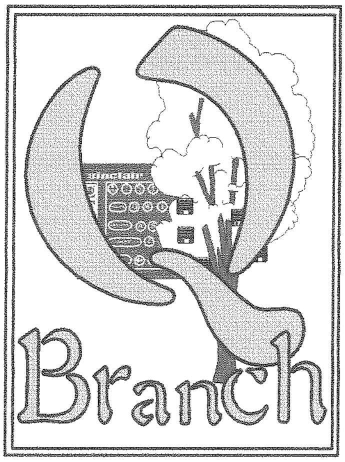
Finally, a request was made to try to sort out emailing facilities, since a recent attempt by traders to invite people to let them know if they read QL Today, Quanta or both and to provide their email addresses etc proved largely costly and unsuccessful. It was noted that costs could be saved if members having email access could be sent information about shows in their area etc, although some concern was expressed concerning the Data Protection Act implications. Robin Barker, the Chairman, said that he maintained an email address list on Quanta's website and appealed to members to use the site to keep their details up to date. Sadly, many members had not done this, or provided incomplete details, so Quanta members - please got to

www.quanta.uni.cc

and check your entries on the email list database on that site!

Readers and show visitors - help us and the organisers of the shows to plan ahead and tell us what you would like: one "main event" every year? Every other year? How many shows would you attend? Remeber, these shows are made for YOU! Please write to QL Today or Quanta.

Q Branch



PROGRAMMING

QD 98	£ 45.00
QD + QBasic	£ 59.00
QD + Qliberator + QBasic	£ 100.00
Qliberator	£ 50.00
Master Spy v 3.3	£ 30.00
QPTR	£ 30.00
Easyptr pt 1 & 2 (together)	£ 30.00
Easyptr pt 3 (C library)	£ 14.00
QMake	£ 15.00
QMon / JMon	£ 22.00
Basic Linker	£ 19.00
DISA 3	£ 31.00
QMenu	£ 14.00

Text 87

£ 79.00

Typset 94	£ 29.00
Fountext 94	£ 39.00
2488 drivers	£ 29.00
Epson ESC/P2 drivers	£ 26.00

Text 87 is the only QDOS / SMSQ wordprocessor capable of handliing the full screen on the Aurora / QXL / QPC systems. New drivers are currently being written.

Q s Spring Sale

The Q Branch Spring sale continues and we still have a few of the second user items mentioned in the last issue.

We Have

3 Full superHermes now only £ 60.00

1 QXL II 8Mb £ 110.00

1 Gold Card £ 60.00

2 Q Plane powered backplane £ 4.00

Don t wait till the Autumn when the branches may be bare !

Call or email for details

'Just Words' by Geoff Wicks

THESAURUS, STYLE CHECK

£ 10.00 ea / ANY 2 PROGRAMS £ 18.00 / ALL 3 PROGRAMS £ 25.00

(Includes Pointer and non-pointer driven versions)

(P.E. versions need Hot_rext, WMAN and PTR_GEN or SMSQ/E to run)

Upgrades from previous versions £ 2.50 + S.A.E. New Manuals £ 1.50

QL2PC

Convert text files from QL to PC formats and much more !

Only £ 10.00 Now with HTML support !

Spelling Crib : PD program £ 1.50 +SAE
or Free if you buy all three programs

UTILITIES

FiFi 2	£ 18.00
QSup	£ 28.00
QSpread v2.04	£ 48.00
Cueshell 2	£ 15.00
Qload / Qref	£ 15.00
Disk Mate 5	£ 16.50
QPAC 1	£ 20.00
QPAC 2	£ 40.00
QTYP 2	£ 30.00
QLQ	£ 28.00

We are currently out of stock of the SuperBasic Reference Manual
Place your order now to get one as soon as it is reprinted

The SBASIC / SuperBASIC Reference Manual

The complete definitive guide to BASIC programming in QDOS / SMSQ including three disks of PD toolkits, example procedures and an electronic index.
compiled by Rich Mellor, Franz Hermann and Peter Jaeger

Over 500 pages !
£ 40.00
+ postage

Tel +44 (0) 1273-386030 fax +44 (0) 1273-381577
Mobile +44 (0) 7836-745501
email qbranch@qbranch.demon.co.uk web : http://www.qbranch.demon.co.uk

Q Branch

Q Branch

Feeling out on a limb ?
Reach out for Q Branch.
Suppliers of Quality QDOS/SMSQ products
Hardware and Software.

20 LOCKS HILL, PORTSLADE,
E. SUSSEX. BN41 2LB. UK.

Please Note our new address and phone numbers

ProWesS

ProWesS (now free !)	£ 1.60
DATAdesign	£ 20.00
Fontutils	£ 28.00
File Search	£ 11.00
PFlist	£ 11.00
Dilwyn's Fontpack	£ Call
LINEdesign v 2.16	£ 22.00
PWfile	£ 17.50

Paragraph

The ProWesS word processor

¶ Demo version £ 1.50 + postage
Full Registered version £ 18.00

Version 2.03 available now !

Hardware

We have a small stock of second user items. Auroras / Qubides / Gold Cards / Qplanes / superHermes etc. call us to get details of the items available. These are going fast so call soon.

QXL II	**£ 100.00**
Recycled superHermes	£ 65.00 *
Recycled Gold Card	£ 50.00 *
Recycled Aurora	£ 75.00 *
Qubide	£ 55.00
Qplane	£ 25.00
Aurora cables	£ 3.00
Aurora rom adaptor	£ 3.00
'Arfa Braquet'	£ 8.00
'Son of Braquet'	£ 18.00
The 'Braquet'	£ 16.00
MC plate	£ 6.50

* when available.

14" and 15" monitors for the Aurora - Call.

SMSQ/E

Gold Card / Atari / QXL Version

£ 69.00

Various Atari versions : call for details

QPC 2 full colour version!

£ 82.00 (£ 67.00 SMSQ/E Owners)

Upgrades from QPC 1 £ 35.00 (return master disk)
Upgrades from QPC 2 v1 £ 30.00
Special offer !

Get Cueshell for only £15.00 with any copy of SMSQ/E or QPC 2 upgrade !

Q Branch Programs

The Knight Safe 3	£ 35.00
upgrades from previous versions	£ 5.00
Q - Route v1.08C Route finding programme	£ 25.00
Q - Count Pointer driven home accounting	£ 25.00

The Fractal Collection !

This is a brand new program which will produce stunning animated fractal patterns. It will run on anything from a Gold Card to the Q40 and will be capable of using the power of the colour drivers when they are released. Complete with many example files and routines to design your own screens.

Only £ 35.00

QPC 2 v2 FULL VERSION AVAILABLE NOW

Qubide upgrades to version 2.01 £ 8.00

VISA

We can accept payment by VISA, Mastercard and Switch. You can also pay by Eurocheques made out in Sterling or a Sterling cheque drawn on a UK Bank. Prices include Post and Packing in Europe.

MasterCard

The QL Bureaucracy

Geoff Wicks

A couple of years ago I made a claim on the British government for the grand sum of £12.90 or about 20 Euros. It took over 27 official documents and forms to process the claim.

A problem of working semi-freelance is preserving your state pension rights, and to be on the safe side I registered for these. I soon discovered the only machinery for doing so was by making a theoretical claim for full social security benefits. The first question I completed was that I was not claiming these, but the bureaucracy still assumed I was.

One letter I received 3 times. It told me I would be taxed on the benefits I was not claiming. It then told me I had not been paid the benefits I was not claiming, and finally told me they had not been able to calculate the tax on the benefits I was not claiming. Dealing with the officials was just as bad. One severely rebuked me because I had failed to name the post office where I would collect the benefits I was not claiming. When I complained about this, I was told the rebuke was in my own interest, otherwise there could have been an undue delay in paying me the benefits I was not claiming.

We all love these stories of official bureaucracy. Indeed, this particular example caused a stir in parliamentary circles, and eventually had the personal attention of the minister.

What has this to do with the QL? Officials are human beings like us, and perhaps we could become just as rigid and bureaucratic in our thinking as them. Sometimes when I look at the QL world I fear we already are.

The officials dealing with me were so busy collecting pieces of paper that they lost sight of the purpose of each piece of paper. They did not realise that the law, although restrictive, had given them far more flexibility than they thought. By concentrating on one small part of their task they missed the total picture.

Some QL-ers, including leading figures who have contributed much to the QL scene, do the same.

Look at the question of speed, which is what inspired this article. Occasionally speed becomes a hot issue in the QL-users email group, often in connection with the QPC versus Q40 debate. It can become quite heated. The number of contributions shoots up considerably with lengthy lists of statistics and debates about which are the most accurate benchmark tests and what are fair comparisons.

Frankly these discussions bore me and I suspect many others. For us the battle between QPC and Q40 is of academic interest. We have admiration for both Marcel Kilgus and Peter Graf, both of whom have achieved something most of us could not do. Both have made an immense contribution to ensuring the future of the QL, and Peter has had to cope with additional logistical problems in the changing availability and prices of components and the production and financing.

For most QL users speed ceased to be a major issue with the arrival of the Gold Card. At one time QL magazines had articles featuring the relative speeds of different types of coding for the same operation. Software and hardware enhancements like Speedscreen and Lightning were necessary to speed up operations. The Turbo compiler, which seems to have been optimised for speed, gained ground from QLiberator in spite of the latter's greater compatibility.

It is interesting to know that my QXL2 is about 25 times faster than the original QL black box, and that QPC2 on my laptop gives roughly the speed of a Super Gold Card, but further than that speed does not interest me.

I am not a Luddite. Indeed it would be hypocritical to be so, because without the speed advances of the Gold Card, Just Words! programs would be too slow to be usable. However for most text applications, including word processing, the QL now works faster than you can think and type. Speed is more important in number crunching or graphics software, and it will gain importance with the new colour drivers, internet access and vector fonts.

It is a similar story with memory. On my bookshelf is a book containing 30 programs for typing into a computer. It has such favourites as Lunar Landing, Duck Shoot, Break in, Calender, Biorhythms and Orbital Invaders. What makes the book special is that it was written for the ZX81, and all the programs fit into 1K. Those were the days when you saved memory by putting numeric variables in a string array and by using n=INT(a/a) instead of n=1.

The first QL gave us a "massive" 128K, but also brought with it a warning. The QL consumed more memory than its predecessors and its 128K was roughly equivalent to the Spectrum's 48K. To be useable the QL needed at least 256K memory expansion, but preferably 512K. The Trump Card increased this to 786K, and the Gold Cards to 2Mb and 4 Mb. Now my QXL2 has 8Mb and my QPC2 can be programmed to at least 16Mb.
What do we do with all this memory? How many of us really need more than the 4Mb of the Super Gold Card? Again this is a situation that could change with the colour drivers, internet access and vector fonts.

Vector fonts?
We have now had these in ProWesS for about 5 years, but you would scarcely realise it. As far as I know there are only 3 commercial programs, LineDesign, Agenda and Paragraph, making use of ProWesS. Most of us, including a loud mea culpa from Just Words!, have never tried programming in ProWesS, but how many of us always load the ProWesS extensions in our boot program? How often do we use these extensions? Do we know what they do?

There are numerous extensions that can be loaded into the QL, but do we look critically at our use of these? Some years ago it was almost essential to install the Turbo Toolkit as many programs, including early versions of Just Words!, used its commands. How many of us ever read or understood the Turbo Toolkit manual? I suspect most of us used only a limited number of commands from the toolkit, but we still loaded almost 8K of extensions in our machines. One of my QL requests is for the Turbo Toolkit to be split up into modules like Simon Goodwin's DIY Toolkit, so that you can "pick and mix" the ones you want. Even better then to incorporate them into programs using QLiberators $$asmb directive.
If you load too many extensions, you run the risk of program crashes caused by compatibility problems. Or you may find your programs will not run on someone else's machine. When I wrote the pointer version of Solvit-Plus, I did it on a QXL SMSQ system, which has the non QDOS command PEEK$. I tested the program for downward compatibility on my JM QL, but failed to notice that my boot disk loaded the Turbo Toolkit which also has a PEEK$ command. I wrongly thought the program was compatible with all systems until after release when customers reported a crash.

Similarly we need to look at our buttons or cascade menus. Some years ago I was criticised by many QL Today readers for writing that one user's buttons reminded me of a general's medals. It produced several interesting articles in reply in which the authors argued that buttons required fewer key presses than loading a program in other ways. The arguments were persuasive, but, in my opinion, missed an essential point. How many of the programs that you have on a button do you use regularly. On my PC my cascade menu is filled with programs I hardly ever use.
Buttons are most effective when used with a mouse, but how critical are we in our use of these? I have the impression the QL community can be divided into lovers and haters of mice. When does a mouse help us and when does it hinder us in our work? Many QL users place a pointer driven word processor high on their QL wish list, but does a mouse make it easier to use a word processor? On PC word processors I find the pointer frequently obscures parts of the text. Using a mouse to mark blocks does not always proceed smoothly. Continually switching from keyboard to mouse is irritating. I think others have these problems, because I frequently see "keyboard shortcuts" published in computer magazines.

One of the QL strengths is multi-tasking (or rather task sharing). I do this more often on the QL than on the PC, but it would be rare for me to have more than 4 programs running in my machine at once. Usually it is Text87 or Perfection with either Just Words! programs or LineDesign. Other users boast of multitasking 12 programs or more. What are these programs and why do they do it?

So, how "bureaucratic" are we in our QL use? Do we need high speeds and massive memories for our daily QL use? How often do we use each extension that we load? It would be an interesting exercise to keep a log of your QL use during a typical week. Note which programs you use and how often. Perhaps the question we should ask is which programs we are going to use on the QL in the next couple of hours, and then load just the minimum requirements for that session.

Another German/Austrian QL Show ... or even "QL 2001"?

Friedemann Oertel is organising a QL Show in one of the nicest areas of Germany: Berchtesgaden! Just a hop over the Austrian border, very close to Salzburg, is it an ideal venue to combine hobby and holidays.

We hope to turn it into "the" event for 2001, as it is of course not just for German and Austrian QLers - everybody is welcome as always! Not too far from Italy, not too far from Switzerland, and definitely worth travelling a longer distance like from France or England if only just for the countryside should bring many QLers together once again, we hope.

The date is October, 5th to 7th. Easy to reach, by car, by train, an hour from Munich away and very, very close to Salzburg (20km) makes it an ideal venue.

The national park on its own is also woth adding an extra day or two, and the weather in October is usually very good, clear view and rather mild.

Apart from the QL show, many activities are available for the rest of the family for bad and good weather - there is the Königssee, the Watzmann (high Mountain), Thermal baths, Saltmine and much more - Friedemann has compiled a long list.

Friedemann has also arranged a set price for accomodation: 85,- DM (about £27) including breakfast and a very rich evening meal. The food in the south of Germany and Austria is very good, as visitors of Austrian shows will confirm. The rooms are equipped with TV, Phone, Shower, Bath etc.

The price is only valid if booked via Friedemann!

Don't wait too long. Last booking needs to be done by 12th of September.

Friedemann Oertel
Grabenweg 24
83471 Schönau am Königssee
EMail: F. Oertel.BGD@t-online.de
Tel. +49-8652-948713 or +49-170-4065272

Getting into QPAC2 - Part 2

Roy Wood

The Boot File Itself

So now we come to the nitty gritty of the problem. How do I write a Boot file ? Well the first thing you should have is the line which calls up the toolkit (unless you have a Super Gold Card and have made that automatic) so our first line should be:

```
100 Tk2_ext
```

Once we have got that over and done with the next lines should open the windows and set the screen. I have always hated the red and white QDOS windows so I start another window over the top and then write a little message into it:

```
110 MODE 4   REMark assume a monitor not a TV.

120 WINDOW #0;512,42,0,214:BORDER #0;1,4,0
130 WINDOW #1;256,140,256,60:BORDER #1;1,255
140 WINDOW #2;256,140,0,60:BORDER #2;1,255
150 OPEN #3;'con_512x150a0x60':BORDER #3;1,
    255:INK #3;4:CLS #3
160 Print#3,"                                   "
170 PRINT#3,"   Q Branch BOOT FILE              "
180 PRINT#3," Prepared for Our Customers        "
```

This gives us a nice black screen with a short message in it. The next job we need to do is to set up a DATA default and a PROGram default. You do not strictly have to do this but it does make the use of some programs much easier. Many programs will look first on the DATA default device when they are asked to load a data file or (even worse in my opinion) look on the DATA default for the data they need to run and, if they do not find it, fail with an error message. The PROGram default device means that you can type: EX myprog and know that, provided the program is executable and in the directory you have previously defined as the PROGram default device, it will run.

```
190 DATA_USE flp1_ : PROG_USE flp1_ :REM
change these devices  to suit your system.
```

Next we need to load the Pointer Environment:

```
200 LRESPR ptr_gen
210 LRESPR wman
220 LRESPR hot_rext
```

Next we load the menu extensions

```
230 LRESPR menu_rext
```

and, of course QPAC 2 itself.

```
240 LRESPR flp1_QPAC2
250 :
260 REMark - Any more resident extensions?
270 REMark - Now Hotkeys to start things off
280 :
```

Now we come to the part where we set up our system and this is where we have to start making some decisions. One of the good things about the way that the QL uses its operating system is that there is nothing that you can do in the Boot file that can make any permanent changes to the operating system itself so you can experiment with the system setup at will. Get one line wrong in the Config.sys or Autoexec.bat of a PC and you can be locked out of the system completely.

```
290 ERT HOT_PICK ('b',''):
    REMark pick superBasic
```

This line creates a HOTKEY which will pick Superbasic if it is buried. If you have a screen full of program windows with Superbasic running underneath and you press 'ALT/b' SuperBasic is immediately picked to the top.

```
300 ERT HOT_THING ('1','Files';'\dFLP1_')
    :REMark create a files thing for flp1_
```

QPAC 2 installs a basic procedure to read directories but, if you want to read a directory of a specific device you have to tell it which device and give it a HOTKEY to call the directory. the line above breaks down into the following components:

ERT	The error trapping keyword
HOT_THING	Create a Thing that can be called
'1'	The HOTKEY itself activated by pressing 'ALT/1'
'Files'	The QPAC 2 procedure used to read the directory
;'\dFLP1_'	The device that has to be read

In the above example the ;'\d part of the line is very important because it tells 'Files' procedure installed by QPAC 2 to create a directory of the device named. If you have a Gold or Super Gold Card or any other memory expansion or system that has level 2 device drivers then this can also be the name of a subdirectory such as ;'\dflp1_DATA_'

```
310 ERT HOT_THING ('2','Files';'\dFLP2_')
:REMark ditto flp2_
```

This is the same as for flp1_ except that you press 'ALT/2' to get the directory of flp2_.

```
320 ERT HOT_THING (CHR$(232),'Button_Sleep')
    :REMark create a 'sleep' Thing.
```

QPAC 2 has a procedure that will put any job to sleep on a button ready to be used again and this line sets that procedure up for you to use. CHR$(232) is the F1 button so, by pressing 'ALT/F1' in any job you will find that job has now become a QPAC 2 button with the job name in it. This will happen with any well behaved program and the more modern ones even put the name of the file you are working on into the button. This is a good way of moving from one program to another without cluttering up the screen of the computer.

```
330 ERT HOT_THING ('.','Button_Pick') :REMark
    create a Thing to pick the button frame
```

This sets up the HOTKEY that will call up the button frame. This is another potentially useful trick which will work from any program that you are using. Press 'ALT/.' and the button frame appears at the chosen position on the screen. You can use this in conjunction with the Stuffer Buffer to call up a files menu, choose a file and the return the name of that file to the program you are using. (see the section on the Stuffer Buffer below)

```
340 ERT HOT_LOAD1 ('x','flp1_Xchange')
    :REMark load Xchange when needed
```

In this boot file we are assuming the lowest possible amount of extra memory so we are not going to load anything as a resident program. Instead we are loading Xchange, the Psion suite, when it is needed using HOT_LOAD1. The '1' at the end of the HOT_LOAD command simply means that, as mentioned above, only one copy of the program will be loaded. The first time you press 'ALT/x' Xchange is loaded ready for use. If you then put this to sleep or bury it under the other windows then pressing 'ALT/x' again will 'Pick' it to the top or 'Wake' it. If you had used the HOT_LOAD command without the '1' then a new version of Xchange would be started every time.

```
350 ERT HOT_LOAD1 ('m',flp1_menuconfig')
    :REMark load menuconfig when needed.
```

Menuconfig is the configuration program for all level 1 and level 2 config blocks. These blocks can be found in a number of different programs, including menuconfig itself. A fuller explanation of this program and config blocks themselves can be found in the supplement to the QPAC 2 manual. For the moment we will only say that this program can be used to set up the various aspects of the programs it is used on. Among the things that can be changed in QPAC 2 are the colours of the program, status of the files menus, the mouse settings, position of the button frame and whether or not the program asks for confirmation of actions. Most pointer programs have these blocks in them and, once you have got used to using them, you will find that they make the use of the programs much easier. Level 2 config blocks store their setting in a file called 'menuconfig_inf'. When you upgrade a program with level 2 blocks and you have already configured the previous version to your satisfaction you need only hit the 'Update' item in the menuconfig box to restore all of your previous settings.

```
360 ERT HOT_WAKE ('P','Pick')
    :REMark call up the Pick Menu
370 ERT_HOT_WAKE ('R','Rjob')
    :REMark call up the Rjob Menu
380 :
```

Now we come to use another feature of QPAC 2, the internal programs. As I said before, once you have loaded QPAC 2 you have access to several programs which make using the QL much easier. We have already set up the Files menus and now we come to set up two of the other programs for use. The first of these is the 'Pick' menu. Since these programs are already loaded and operating within the system there is no need to give a drive name or to use either the HOT_LOAD or HOT_RES commands. All we need to do is to 'Wake them'. I have only used two of the programs here because you can gain access to all of the other QPAC 2 programs from the 'EXEC' menu that we will discuss later. The 'Pick' menu will give you a list of all programs currently running in the computer or set up on HOT_LOAD commands. Moving the pointer to any one of these will immediately call it up. This means that pressing 'ALT/P' in any program will give you a menu that will always take you to any other program in your system. Remember when you had to press 'CTRL/C' to do that? Another thing to note is that I have used the capital letter here. If I had used the lower case letter for the HOTKEY then either 'ALT/p' or 'ALT/P' would have worked. By using the capital letter I have set it up so that only 'ALT/P' will call up the 'Pick' menu and 'ALT/p' can be used for another program. Rjob is the same as the SuperBasic command except that you now have a menu from which to choose which job you want to remove from your system. Very useful if a job has crashed but the system is still operating.

```
390 REMark - Set up, now go
400 :
410 REMark now we create buttons for the screen
420 :
```

```
430 BT_SLEEP 'Pick'
440 BT_SLEEP 'Exec'
450 BT_SLEEP 'Rjob'
```

This next part of the Boot file is concerned with creating the button frame. This is a row of small windows, each of which contains either the name of the program or a title that you you have given it. The first three buttons are for the QPAC 2 programs 'Pick', 'Exec', and 'Rjob'. We have already discussed the first and the last of these but the second item is a very interesting part of the QPAC 2 program. The 'Exec' menu gives access to many of the other programs in the system and can be very useful in manipulating the system itself. There are a collection of menus available from QPAC 2 and these can all be put onto a button by the BT_SLEEP method but this would result in a large number of buttons on the screen most of which you rarely use. In my system I have the three buttons set up here and use the 'Exec' menu to access the others whenever they are needed. For example, there is a menu called 'HOTKEYS' which is a list of all of the HOTKEYS I have set up in the Boot file and what they actually do. Sometimes, if I am adding a program to system, I need to know which ones are still available so this menu comes in very handy. There is another menu called 'Jobs' which gives a list of all of the Jobs running in the computer at the current time. If you click on one of the items on this list you will get a further window giving information on that job. I do not need to have all of this on the screen for most of my daily use of the computer but sometimes it is very useful to be able to see what is happening. The 'Exec' menu is the key to open the door to the works of the computer.

```
460 BT_HOTKEY 'x','Xch'
    :REMark put Xchange on a button
470 BT_HOTKEY 'm','Menuconfig'
    :REMark put menuconfig on a button
480 BT_HOTKEY '1','FLP1'
    :REMark put flp1_ on a button
490 BT_HOTKEY '2','FLP2'
```

The next four lines set up the buttons for the Xchange program, menuconfig and the two files menus. These lines break down as follows:

BT_HOTKEY	The command to create the button itself.
'x'	The HOTKEY that will call the button.
,	The separator (very important).
'Xch'	The name that will appear in the button.

Note that we have used the same HOTKEY to start the button as to HOT_LOAD it. If you give it a different HOTKEY then it will not work since all that is happening here is that when you DO the button it is the same as pressing the 'ALT/x' combination itself. You can put any name that you like in these buttons.

```
500 d$=DATE$:dd$=d$( 10 TO 12)&d$(6 TO 9)&d$(1
    TO 4):REMark create a proper date string
```

This line creates a date string that can be easily used. If you typed 'PRINT dd$' at the command line you would get the date in the following format: **12 Oct 1996**.

```
510 HOT_DO 'b'  :REMark Pick Button Frame
520 HOT_DO CHR$(232) :REMark put the system to
    sleep
```

These two lines introduce the HOT_DO command. When you have set up a HOTKEY you can actually activate that key from Basic using the HOT_DO command in this way. The first of these two lines 'Picks' the button frame to the top and the second puts SuperBasic to sleep on a button. I really like this way of starting the system because it leaves you with a blank screen and a neat row of buttons at the top ready for action. If you want to use the SuperBasic interpreter then just DO the Superbasic button.

```
530 ERT HOT_KEY ('^',dd$) :REMark put the date
    onto a hotkey
540 ERT HOT_KEY ('Q',CHR$(240)&'Q'&CHR$(10))
    :REMark put F3/Quit onto a hotkey
```

These next two lines introduce a different kind of HOT_KEY. in the first the date string that we created earlier is now put onto a HOT_KEY. If you now press 'ALT/^' (ALT/SHIFT/6) the date will be typed into the keyboard queue. In the second of these examples pressing 'ALT/Q' will issue a string of keypresses in a similar way to the macros on modern word processors. In this case they give the following sequence:

CHR$(240)	F3
'Q'	The letter 'Q'
CHR$(10)	ENTER

This is a very useful sequence because many of the Pointer Environment programs use F3 to call up the commands menu, 'Q' to initiate a 'Quit' command and ENTER to complete it. Try this is the 'Files' menus for instance.

```
550 HOT_GO  :REMark start the hotkey system
```

This is the last line in the Boot file and it is very important because it starts the whole HOTKEY system off. If you leave out this line then the system will not work.

The next and final part will list the whole Boot file and explain a few more concepts.

Gee Graphics! (on the QL?) - Part 22

H L Schaaf

How are matrices used for computer graphics?

Remember all those transformation exercises with translation, scaling, rotation, shear, reflection, and projection in GG8, GG12 and GG13? All can be done with matrices!

What's even better is that a sequence of transformations can be made into a single matrix that does the whole job at once, instead of one transformation at a time. Just multiply all the relevant matrices in the desired order and get the one final matrix that 'does it all'.

My challenge is to incorporate matrix techniques onto the QL using "Mathematical Elements for Computer Graphics" by David Rogers to guide me. Chapter 2 covers 2D and Chapter 3 covers 3D. The book explains transformations with 2 by 2 matrices for scaling, shear, rotation and reflection in 2D. Then it goes to 3 by 3 transformation matrices for handling translations and projections in 2D. In Chapter 3 the concepts are taken into 3D with 4 by 4 matrices and perspective transformations.

Rogers starts with a 2 by 2 matrix:

```
( a b )
( c d )
```

and operates on points as vectors of (x y) values or collections of vectors such as a line from point A to point B:

```
( xA  yA )
( xB  yB )
```

or a triangle such as A, B, C:

```
( xA  yA )
( xB  yB )
( xC  yC )
```

In the 2 by 2 transformation matrix above here's what the various scalar variables accomplish:
  a scales the x values relative to the origin
  d scales the y values relative to the origin
  b shears the y values as a function of the x values
  c shears the x values as a function of the y values

First consider the cases where b = c = 0:

When a = d = 1, this is known as the Identity or Unit matrix. Multiplying by the Identity matrix is the same as multiplying by

1.
If d = 1 and a ‹› 1, then x values are multiplied by a.
If a = 1 and d ‹› 1, then y values are multiplied by d.
If d = a › 1, then a 'pure' enlargement occurs.
If 0 ‹ d = a ‹ 1, then a 'pure' compression occurs.
If a = 0 everything collapses onto the y-axis.
If d = 0 everything collapses onto the x-axis.
If both a and d are zero everything collapses onto the origin.
If a ‹ 0 and/or d ‹ 0, then reflection occurs.
If only d is negative the reflection is through the x-axis.
If only a is negative the reflection is through the y-axis.
If both a and d are negative the reflection is through the origin.
The determinant of a 'pure' reflection matrix is -1

Now take the cases when b and c are not zero:

The following three effects are called shear:
If b ‹› 0, then y-values are multiplied by b AND x.
If c ‹› 0, then x-values are multiplied by c AND y.
If b ‹› 0 and c ‹› 0, then effects are combined.

Special cases when a = d = 0:
If b = c = +1, then reflection is through the line y= x
If b = c = -1, then reflection is through the line y= -x
Again the determinant of a 'pure' reflection matrix is -1

Rotation:
For a "pure" rotation of angle theta about the origin use the following values:
a = COS(theta) b = SIN(theta)
c = -SIN(theta) d = COS(theta)
The determinant of a 'pure' rotation matrix is 1.
Two 'pure' reflections combine to form one 'pure' rotation.
The product of the determinants -1 times -1 = 1.

The term 'pure' makes you wonder if there can be 'naughty' situations. The definition given by Rogers is that if the determinant of the transformation matrix is 1 then we have 'pure' rotation. If the determinant is -1 we have 'pure' reflection.

We can imagine 'noughty' cases where zero terms cause trouble. When the determinant is nought we have the ugly singularity. If a, b, c, and d are nought everything disappears!

A sequence of several different transformation matrices can be combined into a single transformation matrix by multiplying the matrices, keeping in mind that the order of multiplying is important!

A collection of points can be operated on as a matrix, so lines and polygons can be transformed with the same transformation matrix that we use for points.

Thus far all operations have been relative to the origin or a line passing through the origin. To move away from the origin we want to translate along the x and/or y axis.

In order to accomplish 2D translations we go to 3 by 3 matrices and use homogeneous coordinates, which we will investigate next time.

P.S. Here's a thought from Doolittle:
M.H. Doolittle (1887)
"Having given the number of instances respectively in which things are thus and so, in which they are thus and not so, in which they are so and not thus, and in which they are neither thus nor so, it is required to eliminate the general quantitative relativity inhering to the mere thingness of the things, and to determine the special quantitative relativity subsisting between the thusness and the soness of the things."

On topic? - does it help explain "Things" in SMSQ/E?

Hard (ware) times

Peter Graf

Here are some comments on Roy Wood's article in the last issue of QL Today. Regarding the discussion on the internet mailing list: It is not true that Phoebus or I told users what they should run their QL software on. I never had that intolerant attitude, and such an insinuation is neither fair nor helpful. Fairness to the QPC writer is especially mentioned, so I would also like to ask for a little fairness to the Q40 defenders. We really need no lessons about tolerance. So far the claims to have the "only answer" for the future of the QL come only from the Windows/QPC side - but not the QL hardware or Q40/Q60 side.
The "ql-users" mailing list is dominated by MS Windows and QPC issues and there had been a lot of praise for QPC. The author called it a "QL" and a "platform all by itself". At this point I felt free to say that the Q40, but not QPC, really is a platform to run different operating systems. And all the interfaces needed to create even more operating systems for the Q40 are accessible, documented, and free. QPC 2 is not a platform to run operating systems on, because there are no such interfaces. It is a Windows program that emulates one single commercial operating system. I have not said that the user should prefer the Q40 because of this, I merely explained the difference. (Of course someone could run another emulator program under SMSQ/E within the QPC emulator, but how much of a "platform" would that be?)
The mentioned discussion in the "ql-users" mailing list was not about how useful a Windows PC or native QL hardware is, because this depends on the interests of every individual user and no general answer can be given.
Later on Roy Wood makes a very astonishing statement that a Windows PC with 700 MHz Athlon and QPC 2 was "appreciably" faster than a Q40. From my own measurements I know that the opposite is true, at least as an overall result. I clearly can say that the Q40 outperforms the mentioned Windows PC with QPC 2 by far. And not only in the already published benchmarks, but also in other tested applications. I have not written the QL benchmark programs and I have not chosen them - I just used them, and everybody can try themself, with the same results.
So why did Roy tell us that the Q40 was slower? Has he a new application benchmark we should add to the list of interesting performance tests? Obviously not. He just saw a screen redraw and felt is was slower on the Q40. I find it remarkable that nowadays the feelings of a single person are supposed to be of higher objectivity than a variety of benchmark programs under well-defined conditions. Especially as this single person is a trader who sells QPC 2 but no longer Q40. A possible expla-

nation for Roy's experience may be that he forgot to enable the copyback cache on his Q40, so he slowed it down. Or differences in the software setup. Or maybe his "exceptionally complex mixing desk" is a rare application than runs slow on a Q40. I don't think so, but the public has not been given the opportunity to check Roys results on their own machines, so I can't tell.

If Roy comes up with a special SMSQ/E task that is faster on his Windows PC (I doubt that!) I could as well come up with another task that is a lot faster on a Q40, let alone a Q60. We could go on and on. This kind of comparison would be even less helpful than the already established benchmarks.

By the way, the Q40 and Q60 suffer from unnecessary disadvantages when their speed is compared with software that doesn't allow the 68020/30/40 CPU instructions to be used. A 68040 CPU has a lot more powerful instructions than a 68000, but if I am not allowed to use them, I must use several 68000 instructions to get the same result. On a Q40 I waste computing speed just to be able to run the same code on QPC 2. This is because QPC 2 can only emulate a plain old 68000 CPU and does not allow for 68020/30/40 instructions or FPU instructions. This is not related to SMSQ/E, but to the processor (or processor emulation).

Roy says he has advice of great relevance for the "upcoming" Q60 design. As he knows well, the Q60 hardware design is not going on, but has been finished long ago and works fine. Unfortunately the Q60 is delayed, but not because of hardware development. One of the reasons is, that there is no support from the QL traders for a series production. Except my own person, there seems to be nobody who would finance it, purchase the parts, manufacture it, do the assembly, the testing, the support. The only help that has been offered from the QL traders is, that Jochen Merz might be willing to sell finished boards, but without offering customer support.

Of course the QL traders have their reasons not to support the Q60, for example the low profit margin, the amount of time it would cost and so on. I don't blame them. I know that it is a lot easier and more profitable to produce and sell emulator software, than QL style mainboards.

If the Q60 becomes a "dead" board as Roy predicts, it is not because of its design, but because of the long lost time and lack of interest. In this fast moving computer world it can be "deadly" to have a complete, working and tested design, but no resources to produce and sell it quickly. And the reasons why the Q60 series production is delayed are not on the technical side.

At this point I would like to ask if there is a reader of QL Today who is interested in producing and selling the Q40 and Q60. The Q60 still comes along with through-hole components so it is not too complicated to get it soldered in small quantities. I have already covered the setup costs for production of electrically tested printed circuit boards, and purchased some of the components like PCB's and VRAMs. Please contact me by email to

pgraf@q40.de

or snail mail to

**Lahnstrasse 32,
D-35239 Steffenberg,
Germany.**

Now some answers to Roy Wood's criticism regarding the Q40 and Q60. In my opinion it would not be a good idea to replace the Q40/Q60 extension bus by PCI. That would add a long period of SMSQ/E software development (maybe years) and make the board much more expensive. As Roy should know about this, I wonder if he really wants a PCI software development. That would waste a lot of important resources while much more important work would have to wait. For simple IO functions like serial and parallel ports, the advantage of PCI versus ISA is small, and the Q60 graphics are directly on the 68060 bus anyway. The required IO cards are in full production, no need for second hand hardware as Roy Wood wrote. The same applies to EDO RAM and AT cases. At the moment, SDRAM is cheaper than EDO RAM, but that doesn't justify the additional time and costs of a Q60 hardware redesign.

It isn't true that the Q40 hardware can't use a bidirectional parallel port, so there is also no need to change this on the Q60. If an appropiate IO card is used, the remaining work is on the software side. It also isn't true that you can't have audio input on the Q40. The Q40 can directly read audio CDs from a CDROM drive attached via the IDE bus. If you additionally need a microphone input, you could think about using an ISA sound card - the rest would be software development again. No important reason to change the hardware design of the Q60.

It also isn't true that the Q40/ Q60 is only for AT cases. Unlike PC mainboards it has been designed for double usage in both AT and ATX cases. Extension slots and mounting holes fit the ATX specifications. For cost reasons, only a DIN keyboard connector is soldered to the board. A PS/2 keyboard connector with plate and cable could be added. But this is not yet necessary. AT cases are cheaper, and it is not true that they are no longer manufactured.

Roy Wood criticises the Q40 and Q60 because of a supposed lack of ISA cards and AT cases, so I wonder why he does not criticise the Goldfire here. If he finds it already difficult to get new ISA cards, I really wonder where and how he wants to find new cards for the special multiplexed QL bus defined by Nasta.

By the way, there has been a fairytale about a Q40 hardware problem that makes individual Q40 boards behave differently. If I remember correctly, this tale had also been published by Roy Wood in QL Today, so I'd like to mention here, that it has ended. It turned out to be an SMSQ/E software bug that initialized a register in an unpredictable way...

Reading articles like those from Peta Jaeger, Wolfgang Uhlig and Roy Wood, I get the impression that nagging and grumbling about the Q40 and Q60 has become the new style of QL Today. Sometimes I think a part of the QL scene has lost orientation. In the Atari scene for example they have spent a million or more to develop a 80 MHz 68060 machine (Milan II, Phenix). A lot of Atari users would be very happy to be offered such a native hardware. In the end the Milan developers failed for technical reasons. And the Phenix was vaporware.

In the QL scene we have got (in our own style) what the Atari has not. Maybe we have the fastest 68K computer there ever was. It's name is Q60, it has been a reality for years and it works. And now? Are we proud? Is there a lot of interest? A lot of noise? Not yet. At least not from the QL traders and the published opinions in QL Today. I get some positive response from some users, and I am thankful for those users who still want QL hardware. But I also hear voices like: "Maybe the Q60 is magnitudes faster under SMSQ/E than anything else. I might be interested, but I could get the latest multimedia Windows PC for the same money."

Hard times for QL hard-ware these days.

QL or QPC

Claus Graf

Firstly this is an answer to the article "Q40 or QPC" by Wolfgang Uhlig (see QLToday Volume 5, Issue 5, Jan/Feb 2001). The second part of the article deals with the situation of the QL in general.

I've been watching the M$ Windows/QPC versus QL/Q40/Q60 debate for a while now. Usually I prefer not to say much about that, because I prefer programming for the QL. But the discussion has reached a point where I cannot be silent anymore.

My article is written and sent to the editor with a Q40. I say this because there are some people who do not cease to claim you need a Windows/ Intel x86 ("Wintel") PC to do normal day work.

In his passionate and emotional article Wolfgang Uhlig sings the praise of Microsoft ("M$") Windows. It is Wolfgang's right to like M$ Windows, of course. But I don't want to read half of a column about how much "pleasure many people" have with M$ programs. There are enough magazines that deal with M$ programs and contain M$ advertisements, so that we don't need that in QL Today also.

He calls people who complain about crashing operating systems "purists, hackers, the QL-ers". Then he tells us that the latter use "stone age instruments". This stands for itself I guess. I cannot believe that a computer or a program is old fashioned or as Wolfgang says a "stone age instrument", just because it's reliable and doesn't crash.

Wolfgang says that "a lot of people must have PC for their work; others like to work with photos, layouts and large complex texts and they cannot manage without". We both know the lack of good programs for QDOS/SMS. Nevertheless you can do a lot with QDOS/SMS programs (there is digicam software and pqiv for photos, for example). If you have a task that you cannot do with QDOS/ SMS, you will probably be more interested in working on a QL software solution, if you have a QL hardware than if you have a Windows PC. Additionally it is not bad to have the opportunity to switch to Linux on QL hardware. QDOS/SMS comes first - but Linux helps me sometimes when I don't have the QL software yet. Q40 combines the three worlds QDOS / SMS / Linux - running on one machine.

Wolfgang says: "I hear repeatedly from the QL side that MS operating systems are bad, but ours are elegant, userfriendly and reliable. A QL-emulator is thus no solution, but it is better to use the original (now the Q40)." Wolfgang "finds it arrogant to think we can cold shoulder the world and retreat into our stand alone QL system". He says it is "unbelievably stupid". It is definitely not "arrogant" not to have or not to like Wintel PCs. It is not "unbelievable stupid" to do things without the Wintel PC. Why do you offend people who don't want a Wintel PC like me? Isn't it enough that you promote M$ Windows and Intel PC?

What I can stand is, if someone likes M$ Windows. But I cannot accept that someone who does not want to use M$ Windows but prefers to have a QL style computer is called "arrogant and unbelievably stupid". This offends QL, GoldCard, Super-GoldCard and Q40 users. They're waiting for an apology. Wolfgang is part of the majority of Windows fans, but nevertheless he should learn to tolerate the views and preferences of the QL (hardware) fans.

Then he talks about the future of the QL and says "If someone not familiar with the QL gets a chance to know it, then it is not because he has bought a Q40. Equally no one will buy a Q40 purely because it can also run Linux. That's just wishful thinking". I know a Q40 user who never knew the QL before. Please tell me at least one single QPC user, that bought QPC although he never knew the QL before!

In the second last paragraph we read "What I have written is absolutely not a defence of windows, but an attempt to correct the perspective". Yes, here I agree. This is not a defence of windows. This is an M$ Windows advertisement!

And finally he writes "..I think only the QPC coupled with new and modern software will create the chance for the QL to survive.." I think the QL cannot survive with QPC - per definitionem. Because the QL is hardware. The QL is a computer. On the other hand QPC is emulator software, and not a computer. It needs another computer with another operating system.

Let me give you a few more examples for differences between a QL and QPC:

- A QL is not an Intel PC nor does it need M$ Windows. QPC needs an Intel PC. QPC needs M$ Windows.
- A QL gives me full control over my machine. QPC depends on what M$ Windows decides to do with my machine.
- A QL runs with QDOS, SMSQ/E or whatever operating system I write. QPC runs only SMSQ/E.
- A QL boots fast in a few seconds and can be switched off. Windows/QPC needs ages to boot,

and cannot just be switched off, but has to run through an annoying startup/shutdown process.

- A QL runs QDOS/SMS directly on my machine. In order to run QPC you need to waste time to get M$ Windows and keep it working, search for working drivers, recover from crashes, maybe soon you need to ask M$ for permission to access Windows after changing some of your hardware, you need to take care not to be attacked by viruses or M$ software that violates your privacy. With a QL you can have fun meanwhile!

So, QPC can run some SMSQ/E software after dealing with M$ Windows. It can be helpful for Windows users, but it cannot ***be*** a QL. If it is not a QL, it cannot save the QL scene, in my opinion.

I understand that some people decide to switch to another platform, like Macintosh or Intel and get themselves a SMSQ/E or QDOS emulator for nostalgia reasons or just to read some QL documents. But for me it's weird to pretend that an Intel computer with M$ Windows operating system could be a QL or that the QL may survive with a Windows software. And it's even more strange, that Wolfgang goes beyond that and claims that a Windows program will create the ***only*** chance for the QL to survive. Isn't such an attitude sectarian?

In order to do the best for the future of the QL one should learn out of the mistakes of the past. In other words, what is the reason of the decline of the QL after the GoldCard days? Beside the general problem of only a few programs available, the appearance of the QXL and QPC suggests some kind of migration path to Windows, which many (of course not all) QL users went:

1) QXL started leading the QLer to the Intel PC and MSDOS. I must admit, at that time there was no other possibility to get hands on a 68040 processor. If Miracle built a real, complete new QL with 68040 at that time, then we wouldn't have lost so many QL users to the Wintel platform.
2) Now the QLer is used to Intel PC and MSDOS, he is ready for M$ Windows and the software emulator QPC.
3) Now the former QLer uses mainly Windows and a little QPC, or even leaves the SMSQ/E world completely.

I think the only chance to survive is to keep the rest of the QLers close to the QL. We need to avoid complete migration to Windows and Intel PC, otherwise the QL will be dead soon. Sometimes I fear it's already too late. QPC gets pushed by the QL traders, while they don't care as much about QL hardware, because they can make a little money easier with QPC software. They don't seem to understand that they dig their own (and the QL's) grave.

If people don't have a Wintel PC they generate pressure to push QL Software development further. Or they start to program themselves. QL/GC/SGC/Q40/Q60 owners can identify themselves with their QL style machine and are proud of their computer. Therefore their commitment leads to new software developments. Of course it is less than in a computer scene with millions of users, but there is something going on at least. At the moment Q40/Q60 users are the driving force on both the hardware and software side. The speed and capabilities of the Q40 speaks for itself and I don't need to tell you again here. Therefore I concentrate on the software issue. Without Q40 there would be no high color driver, no sample sound system, no sound device. These developments are essential for the QL because of the technical progress and especially because of the motivation boost. Beside system software Q40 stimulates development of user applications, for example:

- Image viewer pqiv (JPG, GIF, TIFF, PCX, PNG, PIC, generates backgrounds)
- Image viewer photon
- QSPlayer sound player
- Sox sound conversion program, soundtoy sound player
- Sprited high color sprite editor, produces high color icons and bitmaps
- QL 3D modeller and viewer (Adobe Illustrator and .SVG format output)
- Color chooser
- Q*bert high color game
- high color POVray ray tracer
- high color fractal software
- Linux plus thousands of programs, including Internet access, Web browser, word and image processing, CD burning, mp3, relational database system, etc.

For the ones who like to expand the original QL I hope that GoldFire can be finished. I prefer the concept of a complete computer like Q40/Q60, nevertheless I think the GoldFire card is much more preferable than buying an Intel PC (plus Windows, plus QPC program) for the purpose of using QL software.

So, if you want to save the QL, stick to the QL! May it be QL or GC or SGC or Q40 or Q60 or possibly GoldFire or what other kind of QL may be invented. It is very late, but maybe it is not too late yet. Long live the QL!

Using PRINT_USING

David Denham

Most of us now have and use Toolkit 2 on our systems. But how many of us make full use of the facilities it offers? I don't, and recently had a look at the manual in more detail.
I had never really even appreciated that the PRINT_USING command existed, let alone used it much. I have written SuperBASIC programs in the past for my own business, and now realise that this command would have come in useful for formatting some printed output. At the time I wrote my own rather clumsy routines to do a similar job, but PRINT_USING is now proving very useful.

To quote from the Toolkit 2 manual:
"*PRINT_USING is a fixed format version of the PRINT command:*
`PRINT_USING #channel,format_string, list_of_items_to_print`"

Basically, it works like ordinary PRINT, but you can specify a "format" string which tells the command how to format or lay out the printed output. For example, if you are using floating point variables to store monetary values, it is useful to be able to ensure that a figure like ten pounds and twenty pence is printed in a predictable fixed format such as £10.20 rather than the strange looking 10.2
Control of how the command prints items is by means of special characters embedded in the "format_string." The Toolkit 2 manual describes this string as being a "template" for how to print the list of items that follow.
If any of these characters are found in the "format_string":
`+ - # * , . ! \ ' " $ @`

PRINT_USING starts to perform a certain type of character or number formatting. If the character it came across in the "format_string" was not one of these, it prints the character, so for example `PRINT_USING #1,"1234abcd"` just prints *1234abcd*.

The simplest option is using the `@` symbol. If you need to print one of the special characters in the format string, just precede it with an `@` character:
`PRINT_USING #1,"@$"`

If either the single quote or double quote symbols are met within the string, the text bounded by that symbol is written out "as is" - this is useful for including currency names or abbreviations when printing numeric values, or just simply for printing of simple text.

A `\` symbol acts like a backslash separator in SuperBASIC - it forces the command to print a newline at that point.

The other special characters allow you to determine how values are printed - are they left justified, fixed width, decimal format, exponent/ mantissa format and so on. These are described as "field" options. An item is taken from the print list for each field and printed in the format described by the field. Basically, the first field takes the first parameter after the format string in the command, the second field takes the second item after the format string and so on.

field type	printing format
#####	**strings:** printed left justified in the specified field size, or simply chopped to fit the space **number:** print the integer value right justified
`*****`	print the integer value right justified. If the number of digits of the integer is less than the space indicated by the number of asterisks, the empty part is filled with asterisks.
####.##	probably the most common style, where the number is printed as a decimal number with the specified number of decimal places
`****.**`	like ####.## but any unused spaces on the left of the number filled with asterisks
##,###.##	like ####.## but with commas to separate the thousands value (as most people would write large numbers
`**,***.**`	as ##,###.## but with unused spaces to the left of the number filled with asterisks
-#.####!!!!	print numbers in exponent form, the - denotes an optional sign +#.####!!!! print numbers in exponent form, the + denotes that the form always includes a sign.

Note that the format of the last two versions above is compulsory - it must start with either a + or - character, one hash followed by either a comma or full stop and end with the 4 exclamation marks.
The decimal number fields specified can have a

Bill Richardson is organising a coach from the UK to the Berchtesgaden German/Austrian QL Show (see page 14) So if you want to join in please contact Bill!

CAMBRIDGE Z88

OFFICE/FAX 01494–871319 (EEC) *W.N. Richardson & Co.*

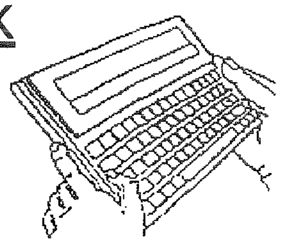
MOBILE: 07808 576118
E-MAIL: WNR@COMPUSERVE.COM

6 Ravensmead
Chalfont-St-Peter
Buckinghamshire SL9 0NB

THE IDEAL PORTABLE COMPANION FOR THE QL, PC AND MAC.

THE CAMBRIDGE Z88 A4 NOTEBOOK

WITH BUILT-IN WORD PROCESSOR, SPREADSHEET, DATABASE, BASIC, CALCULATOR, CLOCK, ALARM, CALENDAR & VT52 TERMINAL.
USES 4XAA ALKALINE CELLS (ca. 20 HOURS) *



WE DO REPAIRS, PART EXCHANGES, AND BUY Z88's & PARTS

NEW! CAMBRIDGE Z88 WITH OZ4+512k Internal Ram £150

ALSO AVAILABLE, 512k RAMPACK £40

1/2 PRICE STARTER PACK
1 - Z88 ☆ PLUS TRANSFER KIT
+ 32K RAM
+ 32K EPROM
£72 ONLY!!

DESCRIPTION	UK	USA
CAMBRIDGE Z88 ISSUE4-128K	**£120**	**$180**
CAMBRIDGE Z88 COMPUTER	**£99**	**$130**
☆**RECONDITIONED Z88**	**£60**	**$90**
32k RAMPACK	£18	$25
128k RAMPACK	£38	$50
32k EPROM PACK	£18	$25
128k EPROM PACK	£24	$33
256k EPROM PACK	£55	$75
EPROM ERASER	£32	$43
PARALLEL PRINTER LEAD	£39	$50
SERIAL PRINTER LEAD	£12	$16
* **MAINS ADAPTOR (230vac; 6v,500ma)**	**£10**	
TOPPER (PROTECTIVE COVER)	£12	$16
CARRYING CASE (PLASTIC)	£10	$12

FILE TRANSFER TO OTHER COMPUTERS

SPECIAL Z88-QL SERIAL LEAD	£10	$15
COPY DISKS OF QUANTA PROGS IMP/EXP & ARCHIVE EXPORT	£2	$3
PCLINK KIT (For PCs)	£25	$35
Z88 TO MAC KIT	£25	$35
Z88 TO BBC KIT	£25	$35

ALL WITH INSTRUCTION BOOKS, CABLES AND EPROMS AS REQUIRED!

POSTAGE UK UP TO £5. EEC £15. USA £20. OTHER COUNTRIES £30.

ALL THE STOCK IS WARRANTIED FOR 90 DAYS. IN THE EVENT OF REPLACEMENT BEING ARGREED, BUT THE ITEM BEING OUT OF STOCK AT THE TIME, A REFUND WILL BE MADE PROVIDED THE ITEM IS RECEIVED IN GOOD CONDITION.

QL & PC COMPUTER USERS WILL FIND THE CAMBRIDGE Z88 ESPECIALLY USEFUL FOR WORK AWAY FROM THE DESKTOP, WITH TRANSFER PROGS DATA CAN BE SAFELY EXCHANGED WITH THEIR DESKTOP SYSTEM.

W.N.RICHARDSON & CO CONTINUES TO PROVIDE FULL SPARES AND SERVICES FOR SINCLAIR COMPUTERS, QL & THE CAMBRIDGE Z88 couple of further options to control how negative numbers are printed, and whether positive numbers have a plus sign. If a field is placed within brackets, negative numbers will be printed enclosed with brackets to denote negative (or debit) items. The use of brackets to denote negative numbers is common practice in accountancy and business circles. PRINT_USING #1,"(###.##)",-1 would print (1.00)

If a - is placed in front of or after the field, negative numbers are preceded or followed by a - symbol
PRINT_USING "-##.##",-1 prints -1.00 whereas
PRINT_USING "##.##-",-1 prints 1.00- which may be more acceptable in a business environment - look at some bank statements which show debits in this format.

Using the - symbol before or after a field means that the number's sign is only printed if the value is negative; if a + symbol is used instead the sign is always preceded by or followed by (depending on where you put the + symbol) the sign whether negative or positive.
PRINT_USING "###.##+",50+50 would print 100+

PRINT_USING has a facility to help users in countries where a comma is used as the decimal separator, e.g. British currency uses the full stop as a decimal point (100.50) whereas some countries would use a comma (100,50). In the examples above, the . character could be replaced by a comma symbol to use that as the decimal separator where commas are preferred.
PRINT_USING "###,##",100 would print 100,00

Obviously, there is some possible scope for confusion between the comma used to separate thousands and hundreds, so PRINT_USING assumes that the last , or . found in the field is used as the decimal separator and the first used as the thousands separator (which may be either a dot or a comma). One further twist is that there is an option to allow a thousands separator to be used without printing a decimal point by placing the decimal point separator at the end of the field (it is not printed if placed at the end of the field):
PRINT_USING "###,###.",999999 prints 999,999

Incidentally, if the number to be printed will not fit into the space allocated by the field specified, you'll get a row of asterisks instead. Try this:
PRINT_USING '##.##',1000
will print five asterisks, as 1000.00 will not fit into the 5 spaces indicated by ##.##

The $ symbol inserted in a field will insert a floating currency symbol which will be printed next to the number rather than to the left of the field size indicated. You might expect
PRINT_USING "$£####.##",10
to print £ 10.00 but in fact it will eliminate the spaces between the currency symbol and the number, giving £10.00. In effect, the currency symbol goes between any spaces padding out the field width and the number itself, so if there is space for 4 digits before the decimal point and the number is two digits long before a decimal point, the currency symbol comes after two spaces, i.e. ‹2 spaces› ‹currency symbol› ‹number›

The conclusion has to be that PRINT_USING is a somewhat fiddly command to use, and the format of the field strings are not easy to remember. Once you have played with it a bit, the command becomes surprisingly powerful, although some careful planning will be needed to get the result you need and it will certainly pay to experiment a little first to find out what is possible.

Here is a short BASIC program for doing currency conversions between Pounds Sterling and U.S. Dollars.

```
100 CLS : CLS #0
110 REPeat convert
120  INPUT #0,'Exchange rate? ';xrate
130  INPUT #0,'Amount £';pounds
140 PRINT_USING "$£####.## = $$####.##\\",
pounds,xrate*pounds
150 END REPeat convert
```

Line 120 gets the exchange rate (for example 1.5) while line 130 asks you to enter the original amount in pounds. The rather fearsome looking line 140 arranges the output of something like
£10.00 = $15.00
The first part $£ arranges that a Pound Sterling currency symbol is placed in front of the original amount of money, the amount is taken from the first parameter after the format string (pounds) and inserted where denoted by the first ####.## part. The short text after the .## is printed 'as is', i.e. the '=' symbol, then the $$ part inserts a U.S. dollar currency symbol, then the second ####.## takes care of formatting the output amount calculated by the last parameter (xrate*pounds). Obviously, if you preferred the dollars and cents separated by commas, you could change the second ####.## to ####,##
As it stands, the program is limited to handling values up to 9,999 pounds or dollars, since my

####.## statements only allow room for 4 digits. PRINT_USING can be used to format printed output on paper too, thanks to QL device independence. Simply open a channel to the printer and use that channel number in a PRINT_USING statement.

Here is a short program to allow you to enter credits and debits and produce a neat listing rather like a statement. Debits are entered as negative numbers, but printed in brackets.

```
100 ch=1
110 INPUT #0,'Opening balance ';balance
120 PRINT #ch,'Opening amount ' TO 60;
130 PRINT_USING #ch,'($'##,###.##)\',balance
140 REPeat loop
150  INPUT #0,'Enter payment name or details ';p$
160  INPUT #0,'Enter credit/debit amount ';amount
170  PRINT #ch,p$ TO 60;
180  PRINT_USING #ch,"($'##,###.##)",amount
190  PRINT TO 70;
200  balance = balance+amount
210  PRINT_USING #ch,"($'##,###.##)\",balance
220 END REPeat loop
```

I hope the article helps to illustrate that despite the initial impression of being a complex command to use, PRINT_USING is one of those useful but not often used little gems hidden in Toolkit 2.

HP Printers and QL's (Part 1)

Peter Fox

Hewlett Packard printers all use a Printer Command Language or 'PCL' and a large part of interfacing the QL with an HP Printer revolves around this point. Most HP Printers embody PCL3 and, by and large, any HP printer which uses PCL3 will behave exactly like another.

However, this will not apply to the more exotic fonts which are occasionally added to HP printers.

What I will deal with in this article is producing drivers for the Psion 4 and also Exchange which will mean that these programmes will output hard copy on an HP printer. In later articles I will explain the detail of these which will make it easier for the reader to write their own drivers.

So far, I have not mentioned Text87 and this is a special case since there are a set of deskjet drivers available which work more widely than the supporting text suggests. This is because most HP printers use PCL3. Using this, you may lose the use of some of the more exotic fonts at the far end of the list but this is, in my opinion, balanced by the fact that one driver will work with most HP printers. A further point with HP printers is that most of them these days need an IBM computer to make them work but will also say that 'Under DOS they use PCL3'. What this means is that you can use any PCL3 printer as a deskjet from either QPC1 or QPC2. It is almost a case of plug it in and it will go, quite surprisingly. The emulation means that you are working sufficiently closely to the IBM system to be able to use the new HP printers to give you hard copy from an HP printer.

In my particular case, I printed this article from Text87 using the Deskjet driver on an HP 1150C Office Jet printer.

The next part of this article involves the use of 'Install_bas' which is the Psion program which allows you to produce new printer drivers. Once this program is run up, you will be presented with a list of printers together with the note that the most recently installed printer was whichever printer you have. Editing the file (using F2) gives you the standard printer commands some of which I will deal with in this article.

All HP printers have parallel interfaces so that the entry for the Port must reflect your hardware. If you are using the Centronics converter that is attached to one of the serial ports, then the entry for the Port will be 'serial'. Since I do not use continuous forms, I have set the lines per page to 66 and the characters per line to 80. In my experience HP printers need CR,LF for the end of line code, not putting in the CR makes the print disappear off the side of the paper and it is not printed out.

There is one special rule when using HP printer Commands, the last letter in the entry must be Upper Case. This tells the printer that the Command stops at this point. This will be important later because it is possible to combine commands which can make the commands shorter but more difficult to decode.

The next two entries in the printer commands are the Preamble and the Postamble. HP use

'ESC'+'E' (decimal equivalents 27,69) as the reset sequence which would be adequate for the Preamble provided the printer defaults into the correct font. The HP reset is a good idea at the end as it puts the printer back into its default state. If you wanted to print envelopes which are done in Landscape orientation on HP printers, the commands to set the printer are:

'ESC'+'&'+'l'+'1'+'O' (decimal equivalents 27,38,108,49,79). Another command that can be set here is the Print Pitch. This uses the command 'ESC'+'('+'s'+'#'+'#'+'H' where the two '#' symbols indicate numeric fields as in '1'+'0' or '1'+'2' for 10 or 12 pitch.

The next four entries cover bold and underline and I am laying these out in a table:

		Decimal Equiv.
Bold On	'ESC'+'('+'s'+'3'+'B'	27,40,115,51,66
Bold Off	'ESC'+'('+'s'+'0'+'B'	27,40,115,48,66
Underline On	'ESC'+'&'+'d'+'1'+'D'	27,38,100,49,68
Underline Off	'ESC'+'&'+'d'+'·'	27,38,100,64

This underline is a single fixed underline but double fixed and single and double floating underlines are also available.

I will not deal with subscripts and superscipts in this article since these require vertical movement plus small characters plus a return vertical movement which will be unworkable unless the commands are combined.

Translates are very personal. I use them to print out '`', '»' and setting the printer to print 12 characters per inch so that you can change the print size within a printer document.

Assembler Part 8

Norman Dunbar

In the last tutorial article, we looked at how easy it was to extend SuperBasic with new procedures and functions. Hopefully you all tried out the homework I left for you to do, if not, there will be points deducted from your final score at the end of the course!

In this article, we shall take a look at the QL's screen memory and how to play around with it. I won't be writing any extensions to SuperBasic this time, but you could extend some of the routines to do so yourselves, and extend SuperBasic to your heart's content.

The Screen

Inside the original QL, there were supposed to be two screen areas. As it turned out, the final product only had one, but some memory was still left around for the second. Unfortunately, the second screen's memory has been partially overwritten by the system variables and so cannot be safely used. To all intents and purposes, we can ignore that second screen and concentrate on the primary screen itself. This is the one we can all use.

Nowadays, we have all sorts of screen modes and resolutions and with the coming of the Q40, we have numerous colours as well. As an old lag, I deal in mode 4 and mode 8 only but as I use a QXL mostly (I am awaiting delivery of QPC 2 even as I type, and hopefully it will have arrived by the time you read this!) I also have more resolution that the old 512 by 256 that the original QL was limited to.

I also have no documentation regarding the resolutions available on other emulators, cards etc so I cannot deal with those here - perhaps someone with more details/knowlege could write a follow up article for an Aurora, Super Gold Card, Q40 etc. (Please!)

In the old days, 512 by 256 was the best you could expect - and only on 4 colours - red, black, green and white. If you wanted more colours, you only had 256 by 256 to play with, however you did get to use blue, yellow, magenta and cyan as well - it was a trade off, as with most things computer related.

OK, here is how it was in the old days .... the screen starts at address $20000 or 131072 in the QL's memory. Each line on the screen, all 256 of them, use 128 bytes to hold the colour information for the pixels in the line. This implies that a QL screen takes up 32K of memory, and indeed this is the case. To get the screen memory address of pixel x,y (x = dots across and y = dots down) a calculation similar to the following was used:

address = 131072 + (y * 128) + INT(x / 4)

This is because each scan line (or row down the screen) starts 128 bytes on from the previous line hence (y * 128). Each row has 512 pixels in it (even in mode 8!) so the dots across are 512/128 = 4. This is why the dots across (or x) must be divided by 4.

DON'T EVER ASSUME THAT THE TWO PARAGRAPHS ABOVE ARE TRUE. The various new cards and graphics modes have chagned all of the above. On my QXL, I can see the screen at the above address only when I run it in QL 512 by 256 mode. The other modes use more memory and in different places, so any program that

writes to the screen at the original addresses will probably cause carnage within the QXL and lead to unexplained crashes later on - if not straight away. It must always be assumed the the old ways have gone forever and we must always calculate the screen start address and how long a scan line is before trying to access the memory.

For those of you who care about these things, the base of the screen address is at offset $32 in the channel definition block, while the size, in bytes, of a scan line is at offset $64. (Except is QDOS version is less than 1.03, in which case, the scan line size is always 128 bytes.)

How to get this information? Easy, given the following code which assumes that A0.L holds a channel id for a scr_ or con_ channel:

```
scr_stuff   moveq   #sd_extop,d0      ; Trap code
            moveq   #-1,d3            ; Timout
            lea     extop,a2          ; Routine to call via sd_extop
            trap    #3                ; Do it
            tst.l   d0                ; OK ?
            bne.s   done              ; No, bale out D1 = A1 = garbage

got_them    move.w  d1,-(a7)          ; Need to check qdos, save scan_line
            moveq   #mt_inf,d0        ; Trap to get qdos version
            trap    #1                ; Get it (no errors)
            move.w  (a7)+,d1          ; Retrieve scan_line value
            andi.l  #$ff00ffff,d2     ; D2 = qdos, mask out the dot in "1.03" etc
            cmpi.l  #$31003034,d2     ; Test for "1x03" where x = don't care
            bcs.s   too_old           ; Less than 1.03 is too old
done        rts                       ; Finished

too_old     move.w  #128,d1           ; Must be 128 bytes
            rts                       ; All done
```

So given that we have a channel id in A0 we can extract the required information from the channel definition block by using the SD_EXTOP trap. This trap takes the address of a routine to call in A2, parameters for the routine in D1, D2 and A1, a channel id in A0 and returns with D1 and A1 holding values returned from the routine called and an error code in D0.

The way we are using it here we don't need any parameters on the way in, but coming out, D1.W holds the scan_line size and A2.L holds the address for the start of the screen memory.

The actual routine itself gets presented with the channel definition block's address in A0, not the channel id. Within the routine we copy the screen base address into A1 and the scan_line size into D1.W and return.

On exit, we need to know if the scan_line size is correct so we call QDOS again to get the version of QDOS in D2. As this corrupts D1 we first save it on the stack. After the trap, D2 holds the ASCII representation of the QDOS version, for example "1.02" or "2.10" or possibly "1m03" for some 'foreign' ROMS (Foreign as in not UK!).

To test for the version we simply mask out the dot or the m or whatever from D2 and if the version is less than 1x03, we simple set D1.W to 128 as this is the only value allowed. All other QDOS versions from 1x03 onwards have the correct scan_line size in D1.W.

So, on exit, A1.L holds the screen address and D1.W holds the scan_line size in bytes. This scan width is useful because we can use it to discover the maximum width of the screen in pixels, provided we know the mode - and I am talking abount mode 4 and 8 only here because that is all I know about!

If we have, as I have on my QXL, a scan_line of 160 bytes, what is this telling me? It says that the number of pixels across the screen will fit into one scan_line of 160 bytes. In mode 4 I know that one word of memory holds the data for 8 individual pixels. In mode 8, I know that one word in memory holds the data for 4 pixels. (Or, as My wife Alison refers to them, 'pixies'.)

As there are 16 bits in a word we can assume correctly that two bits hold the data for mode 4 pixels and 4 bits hold the data for mode 8 pixels. Thus we have 160 bytes times 8 bits and divided by 4 to give 640 pixels across in mode 4. In mode 8 the answer will be 320 BUT the screen width is always the mode 4 width. Only the pixels double up in mode 8, so plotting point 639,0 in mode 8 still works! (or is it 0,639 - I can never remember!)

Our calculation above still works because the memory address of a pixel is now :

screen_base + (y * screen_width) + INT(x / 4)

and this works even on a QXL. We come back to this later.

Mode 4 - screen memory usage

So, as I said above, we have two bits per pixel (or 8 pixels per memory word) in mode 4. How does this work? Mode 4 allows 4 colours, in binary the numbers from 0 to 3 can be represented by two bits. Colours are also represented by 'digits' in that if you add two colours together you get a different colour (ok, contrived link, but bear with me!).

The word in memory is used as follows :

Green byte bits 7 6 5 4 3 2 1 0	Red byte bits 7 6 5 4 3 2 1 0
g g g g g g g g	r r r r r r r r

The colours are as follows:

Colour	GR	Value
black	00	0
red	01	1
green	10	2
white	11	3

So white is represented by all colours mixed together, black by the lack of all colours and red and green by themselves.

In memory we have the green byte and the red byte in each word. The green byte is at an even address while the red byte is at the odd address. We use the corresponding bit in each byte to represent the colour for a single pixel as follows:

Green byte bits 7 6 5 4 3 2 1 0	Red byte bits 7 6 5 4 3 2 1 0
0 0 0 0 1 1 1 1	0 1 0 1 0 1 0 1

Combining the same bits from each byte we get the following table:

Bit	Value	Colour
7	00	black
6	01	red
5	00	black
4	01	red
3	10	green
2	11	white
1	10	red
0	11	white

And that is how it works in mode 4. Ok so we know the screen address (or do we, think about it) and we know how to poke values into the correct location so we can now write directly to the screen can't we? More later, keep those brain cells ticking over for now. There is something I have not yet mentioned.

Mode 8 - screen memory usage

In mode 8 we have 8 different colours. To represent the values 0 to 7 we need at least 3 bits. As there is flashing allowed in mode 4, we need a bit for flash on or flash off as well. 4 bits per pixel is wahr we need and that is what we use.

In this mode, the green byte and the red byte are at the same addresses as in mode 4 with the green being even and the red being odd, but the layout is different. The green byte shares with the flash bit where the green bit is the odd numbered bit (7, 5, 3, 1) and the flash bits are in the even bits (6, 4, 2, 0). A similar arrangement goes on in the red byte with the red bits being even and the blue being odd. So the layout looks like this:

Green byte bits 7 6 5 4 3 2 1 0	Red byte bits 7 6 5 4 3 2 1 0
g f g f g f g f	r b r b r b r b

Again the values for the colours represnt the mixing of the reds, greens and blues - much like colours in nature are just mixes of red, blue and yellow. (Light and inks mix differently and so have different primary colours. In photography, we use red, cyan and magenta!)

The colours are as follows:

Colour	GRB	Value
black	000	0
blue	001	1
red	010	2
magenta	011	3
green	100	4
cyan	101	5
yellow	110	6
white	111	7

So the following bit pattern in mode 8:

Green byte bits 7 6 5 4 3 2 1 0	Red byte bits 7 6 5 4 3 2 1 0
0 0 0 0 0 1 0 1	1 0 0 1 1 1 1 0

Ignoring the flash bits and combining the appropriate bits from each byte we get the following table:

Bits	GRB	Colour
76	010	red
54	001	blue
32	111	black
10	110	yellow

The flash bits are strange. At the beginning of each scan line, the flashing is turned off until such time as a flash bit is set - this turns flashing on until the next flash bit which is set is found. This turns flash off again - so the flash bits act like a toggle turning flash on and off each time a set bit is found. Most books I have read on the subject totally ignore the flash bits after this discussion - I am going to go into it in much more depth. Well that was a lie, I'm not!

That calculation again!

Have you had a good think about caclulating screen addresses for pixels then? Better still, have you thought about the problem I hinted at above? What is the problem then?

If each word of the screen memory holds data for either 8 or 4 pixels, then how can we calculate the correct address for each pixel, because it is (now) obvious that the address for the first 8 pixels in each row will be the same in mode 4 (or 4 pixels in mode 8) so our wonderful calculation above needs a bit of tweaking to make it work correctly.

In mode 4, the screen address changes every 8 pixels across. So where x is 0 to 7, the screen address is the same, for x = 8 to 15 it is the next word of memory and so on. The word that the x pixel lives in is found by the calculation, but the actual pixel within that group of 8 pixels is not found. Follow?

Assume row zero and pixel 2, this gives screen address =

base address + (0 * scan width) + INT(2 / 4)

or

base address + 0 + 0

or

base address

This is the same address for pixel 0 through pixel 7. For pixels 8 to 15 it will be:

base address + (0 * scan width) + INT(8 / 4)

or

base address + 2

so we know the memory word, but not the actual bits within it. Remember bits 7 = pixel 0, bit 6 = pixel 1 and so on down (up?) to bit 0 for pixel 7. How do we get to a value between 0 and 7 from any x value? If we AND the x value with 7 that will give us a value between 0 and 7 won't it - lets see:

X	X AND 7
0	0
1	1
2	2
3	3
4	4
5	5
6	6
7	7
8	0
9	1
10	2
11	3

and so on. Are these the correct values for the bits in the word that we want? Try this:

X AND 7	Correct bit
0	7
1	6
2	5
3	4
4	3
5	2
6	1
7	0

Not quite it would appear, but we could always subtract (x AND 7) from 7 couldn't we? That would give the correct answer. So a solution is at hand. If we subtract the result of (x AND 7) from 7, we get the correct bit number in each byte of the calculated memory word. Yippee (or is it - read on.)

Not quite, I'm afraid. If we have the memory address, we can extract the current contents - we must preserve the other 7 pixels when we plot this one remember - so we need to mask out the same bit in each byte of the screen word. If we used the subtraction method identified above, we would needs bucket loads of testing and masking to figure out which bit is required. We need another method. Before we get to that, how exactly shall we preserve the current pixels?

Remember that a pixel is defined by a single bit in the green byte and the corresponding bit in the red byte of the screen word. To set a pixel we must first set its two bits to zero (or black) and then set the two bits according to the requested colour. This turns out to be quite simple.

First create a mask where the bit to be changed in the red and green bytes are set to zero and every other bit is set to 1. If we AND this mask word with the screen word we effectively set that one pixel to black. So far so good. Next set a new mask where the single bit in each byte is the requested green or red bit and all the rest are zero. If we now OR this word with the screen word we have set the pixel to our requested colour. Too many words, lets have an example.

Our screen shows the following colours in the first 8 pixels:

red green green black black white red white

This means that we have the following two bit values for each pixel:

```
01 10 10 00 00 11 01 11
```

Which means that we have the following word in memory :

```
01100101 10000111 = $6587
```

Now let us assume that we want to colour the first pixel (currently red) to white. So our mask to clear that bit (bit 7 in each byte) needs to be set to

```
01111111 01111111 = $7f7f
```

Now we AND this word with the screen word to get the following :

```
01100101 10000111 = $6587
01111111 01111111 = $7f7f
-----------------
01100101 00000111 = $6507
```

Note now that the first pixel has been set to 00 (bit 7 from both bytes) so it has effectively been set to black.

Next we need a white pixel so the colour mask for white must have a 1 in bit 7 of each byte. The rest must be zero to preserve the current colours of all the other pixels. Our mask must be:

```
10000000 10000000 = $8080
```

So if we now OR this into the (new) screen word we get the following:

```
01100101 00000111 = $6507
10000000 10000000 = $8080
-----------------
11100101 10000111 = $E587
```

Taking all the bits into colour values we get this:

```
11 10 10 00 00 11 01 11
```

which translates back to the following colours:

white green green black black white red white

Success, we have preserved all other pixels and set the first one to white. Now we know how to do it to one pixel, it is the same for all the other 7, but the masks need to be changed for each pixel. How?

If we decide to change pixel 0 (as above) the masks are $7f7f and $8080. This is easy. If we want pixel 1 to be changed the masks are rotated one bit to the right becoming $bfbf and 4040 and so on. Look again at our table above where we show the result of (x AND 7) and the correct bit in the screen word - notice that if we assume that pixel 0 is being changed we can rotate the masks by (x AND 7) bits to get the correct masks for whichever pixel we try to set, as the following table shows:

Pixel	(x AND 7)	AND Mask	OR Mask
0	0	01111111	10000000
1	1	10111111	01000000
2	2	11011111	00100000
3	3	11101111	00010000
4	4	11110111	00001000
5	5	11111011	00000100
6	6	11111101	00000010
7	7	11111110	00000001

I have only shown one byte of each mask, the other byte is identical - this should allow Dilwyn to get the table into the magazine!

Looking at the table, we see that the result of (X AND 7) is the pixel we need to set in the screen. If we start with a mask suitable for pixel 0 and ROTATE it to the right by (x AND 7) bits, we get the correct mask for that pixel. This also works for our colour mask as well. Things sometimes become clear when you switch to binary, especially in graphics situations!

We now have the basics for a mode 4 'pixel setting' routine. Lets try it out.

Assume that we want to set the colour of any pixel on the screen to any of the 4 colours we want in mode 4. We can actually use any of the mode 8 colours because only bits 2 and 1 will be used. This means that a mode 8 colour of blue (value 001) will result in a mode 4 value black (value 00) being set for the appropriate pixel. This is exactly how SuperBasic would handle it.

We will use the registers as follows:

```
    D1.W = x (across)
    D2.W = y (down)
    D3.W = colour (0 to 7)
```

```
*-------------------------------------------------------------------------------
* In D3 bit 2 is green and bit 1 is red, we don't need any other bits, so get
* rid of them now. Then shift the Green bit into bit 15 of D4 and the red into
* bit 7 of D3 ...
*-------------------------------------------------------------------------------
start       bra     plot_init       ; Call here (start + 4) to initialise things
```

```
plot_4      bsr.s     calc            ; Get A1 = screen address
            andi.w    #6,d3           ; D3 = 00000000 00000GR0 (showing all bits)
            lsl.w     #6,d3           ; D3 = 0000000G R0000000
            move.w    d3,d4           ; D4 = 0000000G R0000000
            lsl.w     #7,d4           ; D4 = GR000000 00000000
            or.w      d4,d3           ; D3 = GR00000G R0000000
            andi.w    #$8080,d3       ; D3 = G0000000 R0000000 (keep both bits 7)

*------------------------------------------------------------------------------
* D3.W is now set to a colour mask for pixel 0. This is where we want to start.
* Now we need to build a mask to clear out pixel 0 as well. This is easy - use the
* value from the table above. Then we can start rotating them into the correct
* position as detailed above.
*------------------------------------------------------------------------------
            move.w    #$7f7f,d2       ; AND mask = 10000000 10000000
            andi.w    #7,d1           ; (x AND 7) in d1
            ror.w     d1,d2           ; Build correct AND mask
            ror.w     d1,d3           ; Build correct OR mask (colour)
            and.w     d2,(a1)         ; AND out the changing pixel
            or.w      d3,(a1)         ; OR in the (new) colour
            moveq     #0,d0           ; No errors
            rts                       ; All done

*------------------------------------------------------------------------------
* Calculate the screen address for the x and y values passed in D1 and D2.
* Trashes A1, D4 and D5.
* The routine plot_init must have been called to initialise the screen addresses
* and scan line widths BEFORE calling this routine.
*------------------------------------------------------------------------------
calc        lea       scr_base,a1     ; Where we hold the screen base address
            move.l    (a1)+,d0        ; Fetch the screen base address
            move.w    (a1),d6         ; And the scan line size
            movea.l   d0,a1           ; Get the screen base where we want it

*------------------------------------------------------------------------------
* D1.W = x across value
* D2.W = y down value
* D3.W = ink colour required
* D6.W = scan line size
* A1.L = screen base address
*------------------------------------------------------------------------------
            move.w    d2,d5           ; Copy y value (down)
            ext.l     d5              ; We get a long result next ...
            mulu      d6,d5           ; Multiply by scan_line size
            adda.l    d5,a1           ; A1 = correct scan line address

            move.w    d1,d4           ; Copy x value
            lsr.w     #2,d4           ; D4 = INT(x / 4)
            bclr      #0,d4           ; Make even = green byte in scan_line
            adda.w    d4,a1           ; A1 = correct screen word address
            rts                       ; Done

*------------------------------------------------------------------------------
* This routine must be called once before using the above plot routines. It
* initialises the screen base address and scan line width from the channel
* definition block for SuperBasic channel #0.
*------------------------------------------------------------------------------
plot_init   suba.l    a0,a0           ; Channel id for #0 is always 0
            lea       scr_base,a1     ; Parameter passed to extop routine
            lea       extop,a2        ; Actual routine to call
            moveq     #sd_extop,d0    ; Trap code
            moveq     #-1,d3          ; Timout
            trap      #3              ; Do it
            tst.l     d0              ; OK ?
            bne.s     done            ; No, bale out D1 = A1 = garbage

got_them    move.w    d1,-(a7)        ; Need to check qdos, save scan_line
            moveq     #mt_inf,d0      ; Trap to get qdos version (preseves A1)
            trap      #1              ; Get it (no errors)
            move.w    (a7)+,d1        ; Retrieve scan_line value
            andi.l    #$ff00ffff,d2   ; D2 = qdos, mask out the dot in "1.03" etc
```

JOCHEN MERZ SOFTWARE

Im stillen Winkel 12 D-47169 Duisburg
Tel. 0203 502011 Fax 0203 502012
http://smsq.j-m-s.com SMSQ@j-m-s.com

QSpread 2001

NEW

As QSpread is one of the programs most used by JMS himself, a number of changes were more than overdue. We have added a number of useful features which make using QSpread even better:

Apart from some fine-tuning (the print icon now prints the currently marked block or page), export suggests a constructed filename to save you typing, string input appear near cursor etc. we have added a couple of other things:

Password protection. You can now protect the sheets with a password which is requested next time somebody (or you) tries to load the sheet. As we had to build it on existing structures, the protection is not strong, but it will protect it against people peeking at your sheets.

One of the best new features is the context menu. Right-clicking the sheet will pop up the context menu, and it is so useful once you get used to it: the default item will be "mark right end of block", but other options are, for example, number format, justification, set column width and, of course, all the useful grid manipulation commands for inserting and deleting rows and columns. The context menu works on the current block (if clicked into a currently marked block), or on the current cell (if clicked outside of a block). Marking a block can now be done by holding the SHIFT key while you click with the mouse. You'll find it more than useful!

Upgrade from QSpread 99	EUR 15 - £10
Upgrade from older QSpread versions	EUR 30 - £20
QSpread 2001 - New version	EUR 65 - £43.33

TERMS OF PAYMENT

Postage and package **[Germany]** DM 8,99 (if total value of goods is up to DM 50,- then only DM 5,99).
[Europe] £4,50 (if total value of goods is under £15 then only £3).
[Overseas] between US$7.50 (1 item) and US$17.50 (maximum).
All prices incl. 16% V.A.T. (can be deducted for orders from non-EU-countries).
Cheques in DM, EURO, Eurocheques and Credit Cards accepted.
Please note: Prices are based on an average exchange rate of £1 ranging between DM 3,10 and DM 3,20. Prices may be adjusted in case the exchange rate falls out of this range - in both directions!
We can charge your credit cards in £'s, US$, EURO or in DM - please state the currency you prefer.
Please do not send any UK bank cheques in £ - our banks have increased the fee for handling them by 600% (no joke!) so we cannot accept them anymore, unless you add £6 for clearing the cheque. E&OE.

VISA MasterCard

```
            cmpi.l   #$31003034,d2    ; Test for "1?03" where ? = don't care
            bcs.s    too_old          ; Less than 1.03 is too old

save        move.w   d1,(a1)          ; Store the scan_line size

done        rts                       ; Finished

too_old     move.w   #128,d1          ; Must be 128 bytes
            bra.s    save             ; Save D1 and exit

extop       move.l   $32(a0),(a1)+    ; Fetch the scan_line length & store it
            move.w   $64(a0),d1       ; Fetch the screen base - don't store it
            moveq    #0,d0            ; No errors
            rts                       ; done

*-------------------------------------------------------------------------------
* Set aside some storage space to hold the screen base and scan_line width. This
* saves having to calculate it every time we plot a single pixel.
*-------------------------------------------------------------------------------
scr_base    ds.l     1
scan_line   ds.w     1
```

And that is the end of the code. To use the above in your assembly language programs simply call plot_init once to set up the screen base and scan line widths, then call plot_4 as often as you like. Easy stuff.

To test this code out from SuperBasic, ALCHP (or RESPR) some heap and LBYTES the code file to that address and CALL it. This initialises the system by calling plot_init. Now, simply CALL address, x, y, colour and the points will be plotted. Make sure you are in mode 4 or the results may be a bit crazy! An example program follows:

```
1000  PLOT_INIT = RESPR(256): REMark Enough space for plot_8 as well!
1005  PLOT_4 = PLOT_INIT + 4
1010  LBYTES flp1_plot_bin, PLOT_INIT
1015  CALL PLOT_INIT
1020  FOR across = 0 to 100
1025    FOR down = 0 to 100
1030      CALL PLOT_4, across, down, RND(0 to 7)
1035    END FOR down
1040  END FOR across
```

Problems

Ok, so what, if anything is wrong with the plot_4 routine? The answer is that there is no checking to see if the x and y values are out of range. If you try to plot say pixel 2000, 494 the chances are that it would corrupt something in memory (probably a system variable) with either immediate or later results.

It is probably easy to check the x value (or across) because there are 8 pixels per word in mode 4 so multiplying the scan line width (in bytes) by 4 should give the maximum resolution across. Indeed, on my QXL, this works out. My scan line is 160 bytes and the maximum resolution is 640 across by 480 down. 160 times 4 is indeed 640. Unfortunately, I cannot think or find a method of calculating the maximum display resolution in the 'downward' direction.

It may be true that all current display resolutions that are 640 accross must be 480 down, but is this true or not? It appears not. A quick check with the demo version of QPC 2 (an old demo version at that) shows that It can have the following resolutions (across by down):

X	Y
512	256
640	400
640	480
800	600
1024	768
1152	864
1280	1024
1600	1200

So we can already see that detecting a 640 pixels across resolution leads to a decision about the downward resolution, is it 400 or 480?

I feel the need to be told if there is a way, simple and effective and which works on all machines, whether thay are black box QLs or Q40s or emulators, to tell the maximum screen resolution. Anyone got any ideas? If so, Dilwyn will be glad to print the article you are about to write!!

Homework

For this article's homework, I want you to write a mode 8 plot routine in a manner similar to the plot_4 routine shown above. Here are some hints:

1. Avoid the flash bits like the plague. Simply mask them out and set them to zero.
2. The calc routine works for mode 8 as well. No need to change it.
3. The mask for pixel 0's colour needs to be GF000000 RB000000.
4. The mask to clear pixel 0 needs to be 01111111 00111111 ($7f3f).

The algorithm is as follows:

1. Calculate the screen address by calling calc. Sets A1 = screen address.
2. Mask out all but bits 0, 1 and 2 of D3.W This is the pixel colour. D3 = GRB.
3. Shift D3.W LEFT by 6 bits.
4. Copy D3.W to D4.W
5. Shift D4 left by 7 bits.
6. ANDI.W D4.W with $8000 to preserve only bit 15 = G.
7. ANDI.W D3.W with $C0 to zero the G bit currently in bit 8.
8. OR.W D4 into D3 to give the correct colour mask for pixel 0.
9. ANDI.W d1 with 6 to get the correct number of rotates (6 makes it even which it must be because we need to rotate two bits for each pixel.)
10. Rotate right, the two mask words, the correct number of bits.
11. AND.W the mask with the screen word.
12. OR.W the colour mask with the screen word.
13. Clear D0 and return.

The results of (x and 6) are as follows:

X	X AND 6
0	0
1	0
2	2
3	2
4	4
5	4
6	6
7	6
8	0
9	0
10	2
11	2

And so on. Because we are using two bits of the green and red bytes to represent our colour, we need to always rotate by an even number.

To test it all out, add the code to the end of the original file which has plot_4 in it and change the first two lines from this:

```
start       bra       plot_init
plot_4      bsr.s     calc
```

to the following:

```
start       bra       plot_init
plot_4      bra       plot_4
plot_8      bra       plot_8
```

This means that plot_init is the start address, plot_4 is at address + 4 and plot_8 has been inserted at start address + 8, as follows:

```
1000 PLOT_INIT = RESPR(256):
     REMark Enough space for plot_8 as well!
1005 PLOT_4 = PLOT_INIT + 4
1010 PLOT_8 = PLOT_INIT + 8
1010 LBYTES flp1_plot_bin, PLOT_INIT
1015 CALL PLOT_INIT
```

Have fun!

Programming ProWesS in SBASIC - Part 7

Wolfgang Lenerz

Last time, we started looking at the way objects are laid out within the windows. I don't know whether you tried to do as I suggested and attempted to set the two info items beneath the loose items, in such a way that info2 is underneath info1. If not, here is the solution, particularly in lines 370 and 380:

```
100 set_windows
110 test1
115 :
120 DEFine PROCedure test1
130 LOCal object, hit%,hits,dos,times$,mhit$,mdo$
140 LOCal loop%,add_info
150 :
160 : REMark first initialise some variables
170   mhit$="you have hit the item ":times$=" times":hits=0
180   mem=0:object=0:hit%=0
190   mdo$="you have done the item ":dos=0
200 :
210 : REMark now make some strings, note the chr$(0) at the end!
220 :
```

```
230   my_hit$=mhit$& hits&times$&CHR$(0)
240   my_do$=mdo$&dos&times$&CHR$(0)
250 :
260 REMark now create the outline object
270 :
280   outl=PWcreate(0,PW('TYPE_OUTLINE'),PW('OUTLINE_QUIT'),
                   PW('OUTLINE_SLEEP'))
290 :
300 REMark now create the item objects
310 :
320   item1=PWcreate(outl,PW('TYPE_LOOSE_ITEM'),
            PW('LOOSE_TEXT_COPY'),"Hit or do me",
            PW('LOOSE_ACTION_DO'),DO_ROUTINE,
            PW('LOOSE_ACTION_HIT'),HIT_ROUTINE)
330   item2=PWcreate(outl,PW('TYPE_LOOSE_ITEM'),
            PW('LOOSE_TEXT_COPY'),"hitting or doing me will do
            nothing",
            PW('POSITION_RIGHT_OF'),item1)
340 :
350 REMark now we create two infostring objects
360 :
370   info1=PWcreate(outl,PW('TYPE_INFOSTRING'),
            PW('INFOSTRING_TEXT'),my_hit$,
            PW('INFOSTRING_AUTOSIZE'),0
            PW('POSITION_NEXT_ROW))
380   info2=PWcreate(outl,PW('TYPE_INFOSTRING'),
            PW('INFOSTRING_TEXT'),my_do$,
            PW('POSITOIN_BELOW'),info1)
390 :
400 REMark the main loop
410 :
420   REPeat loop%
430     mem=PWactivate(outl,mem,object,add_info,hit%)
440     IF NOT mem:EXIT loop%    : REMark if mem is returned
        as 0, we quit the window
450     SELect ON object
460       =item1
470         SELect ON hit%
480           =0:hits=hits+1
490              my_hit$=mhit$&hits&times$&CHR$(0)
500              PWchange info1,PW('INFOSTRING_TEXT'),my_hit$
510           =1:dos=dos+1
520              my_do$=mdo$&dos&times$&CHR$(0)
530              PWchange info2,PW('INFOSTRING_TEXT'),my_do$
540         END SELect
550     END SELect
560   END REPeat loop%
570 :
580   PWremove outl
590 END DEFine test1
600 :
610 DEFine PROCedure set_windows
620 LOCal not_compiled,upper%,xo%,yo%,xs%,ys%,ysize_0%
630   not_compiled=IS_OPEN(#0)      : REMark window#0 open?
640   IF not_compiled
650     xs%=0:ys%=0:xo%=0:yo%=0
660     PWscrsize#0,xs%,ys%,xo%,yo%
670     upper%=28:ysize_0%=50
680     PWoutln#0,xs%,ys%-upper%,xo%,upper%+yo%
690     WINDOW#0,xs%,ysize_0%,xo%,ys%-ysize_0%
700        WINDOW#1,xs%    DIV    2,ys%-upper%-ysize_0%,xo%+(xs%
           DIV 2),yo%+upper%
710     WINDOW#2,xs% DIV 2,ys%-upper%-ysize_0%,xo%,yo%+upper%
720     BORDER#1,1,255:BORDER#2,1,255
730     PAPER#1,2:PAPER#2,7
740     INK#1,7:INK#2,2
750     CLS#0:CLS#1:CLS#2
760   END IF
770 END DEFine set_windows
```

Another - and sometimes easier way - way to achieve the exact positioning of your objects within the windows are the "glue" and the "direction" types. Let's start with the latter.

The direction type

The handbook tells us that a "direction is a region object which can be used as a container to put other regions inside" - which, whilst it is exact, doesn't tell us that much.

However, remember the first attempts we used last time when trying to put the item2 object to the right of item1, using the

PW('OPSITION_RIGHT_OF')

tag. This had the unwelcome consequence, that the next objects were also put to the right.

As explained last time, Prowess thinks in columns and rows: each new object is put underneath the previous one (vertical - row) or to the right of the previous one (horizontal - column), and the

PW('POSITION_RIGHT_OF')

tag changes the way objects are laid out, so that now they are laid out horizontally. To achieve finally the goal we had fixed ourselves (the two loose items next to each other, the two info items underneath), we had to use quite some positioning tags.

I'm sorry to say this now, after having dragged you through the last installment, but we could immediately have achieved this much easier (but I wouldn't have been able to explain why).

What we need, is a kind of objects that somehow enables use to change direction "locally", without changing direction "globally". this is where the direction object comes in. A direction object doesn't do ANYTHING. It just is there and serves as a "container". You can put other objects inside the direction object, and the way the objects are laid out inside this container (horizontally/vertically) will not influence the way the objects are laid out in the rest of the window. The use of this type can thus be necessary if you want to control the positioning of the objects in the window exactly, like in our example.

To recap, we want to create 4 objects: two loose menu items, two info objects. If we don't use any positioning tags, they will be created underneath each other, the first loose item being underneath the separator. If we use "*PW('POSITION_RIGHT_OF')*" on the second loose item, the info objects will also be laid out horizontally, which isn't what we want at all.

So what we'll do is the following: Create a direction. this, again, will be created under the separator. Create the loose items with the direction as owner. Then create the info objects. The info objects will quite normally be created UNDERNEATH the direction object, in the normal manner.

To achive this, first create a line 315:

```
dir1=PWcreate(outl,PW('type_direction'))
```

Then change lines 320 to 380 in the above example project, so that they read as follows:

```
320 item1=PWcreate(dir1,PW('TYPE_LOOSE_ITEM'),
    PW('LOOSE_TEXT_COPY'),"Hit or do me",
    PW('LOOSE_ACTION_DO'),DO_ROUTINE,
    PW('LOOSE_ACTION_HIT'),HIT_ROUTINE)
330 item2=PWcreate(dir1,PW('TYPE_LOOSE_ITEM'),
    PW('LOOSE_TEXT_COPY'),"hitting or doing me will do nothing")

370 info1=PWcreate(outl,PW('TYPE_INFOSTRING'),
    PW('INFOSTRING_TEXT'),my_hit$, PW('INFOSTRING_AUTOSIZE'),0)
380 info2=PWcreate(outl,PW('TYPE_INFOSTRING'),
    PW('INFOSTRING_TEXT'),my_do$)
```

Run the resulting program - we get exactly what we wanted: item2 next to item1, and the info objects underneath.

Ok, you will have noticed that we created a new line 315, where we set up the direction object. Then, in lines 320 and

330, we create the loose menu item objects but, contrary to what was done earlier, we specify that the owner of the objects is the freshly created direction objects, not the outline. Then we create the info objects normally.

We didn't need to use any positioning or direction (row/column) changing tag, just the direction object. Prowess is clever enough to think that in such a direction object, we would like to change direction compared to the main window, and does that automatically.

The glue type

According to the manual, the "Glue type (which is in fact the same as a direction, but the name better fits the purpose) can be used to create some "special" effects for positioning objects. Glue is an invisible region which can be used for spacing out other objects. For example, if you have two objects of which one should appear at the left of the window and the other one at the right, then you can put some glue in between, and that glue will stretch to eat unused space (according to the scale factor)". Alright, let's try that out. We'll leave the two loose items as they are, but now we want the two info objects underneath, info object 1 under item1, left-justified and info object 2 under item2, right justified. To do that, we'll have to amend our program somewhat, also to make sure that the texts for the loose items are longer and those for the info objects are shorter, to better observe the effect of this invisible glue type.

To keep things simple, I've included the relevant lines from the example program again, notice how I lengthen the strings in lines 320 and 330 and shorten them in lines 170 and 190.

```
160 : REMark first initialise some variables
170   mhit$="hit:":times$=" x":hits=0
180   mem=0:object=0:hit%=0
190   mdo$="done ":dos=0
200 :
210 : REMark now make some strings, note the chr$(0) at the end!
220 :
230   my_hit$=mhit$& hits&times$&CHR$(0)
240   my_do$=mdo$&dos&times$&CHR$(0)
250 :
260 REMark now create the outline object
270 :
280   outl=PWcreate(0,PW('TYPE_OUTLINE'),
               PW('OUTLINE_QUIT'),
               PW('OUTLINE_SLEEP'))
290 :
300 REMark now create the item objects
310 :
315 dir1=PWcreate(outl,PW('type_direction'))
320   item1=PWceate(dir1,PW('TYPE_LOOSE_ITEM'),
               PW('LOOSE_TEXT_COPY'),
               "Hit or do me - oh please oh please oh please",
               PW('LOOSE_ACTION_DO'),DO_ROUTINE,
               PW('LOOSE_ACTION_HIT'),HIT_ROUTINE)
330   item2=PWcreate(dir1,PW('TYPE_LOOSE_ITEM'),
               PW('LOOSE_TEXT_COPY'),
               "hitting or doing me will do nothing -
               unfortunately......")
340 :
345   dir2=PWcreate(outl,PW('type_direction'))
350 REMark now we create two infostring objects
360 :
370   info1=PWcreate(dir2,PW('TYPE_INFOSTRING'),
               PW('INFOSTRING_TEXT'),my_hit$,
               PW('INFOSTRING_AUTOSIZE'),0,
               PW('scale_factor'),0)
375 REMark g1=PWcreate(dir2,PW('type_glue'))
380   info2=PWcreate(dir2,PW('TYPE_INFOSTRING'),
               PW('INFOSTRING_TEXT'),my_do$,PW('scale_factor'),0)
390 :
```

I've also created a second direction which contains the two info objects, to make their horizontal positioning easier.

Run the program as is. The two info objects are right next to each other - that's not what we want. Then remove the REMark from line 375, to create the glue and re-run the program. The difference between the two should be quite clear - when the glue is used, the effect is as described in the manual, one info object is left justified, the other is right justified.

The scale factor

The eagle-eyed amongst you will have noted the use of the "PW('SCALE_FACTOR')" tag for the two info objects. What is a scale factor? Put simply, when putting a certain number of items in a single row, such as the two info objects, PROWESS will grant each object the same amount of space to be drawn in. That seems logicial, that way you get evenly spaced objects, which is what one mostly wants.

You can, however determine that one object should take, for example, twice the amount of screen real estate than the other, by modifying the scale factor for the objects. If you provide for a scale factor of 0, each object is only allocated so much space as it needs to draw itself.

Let's try this out. Put the remark back into line 375, or take this line out entirely: it, and the glue object, have served their purpose. Run the program again. Since the two info objects have a scale factor of 0, info object 2 is drawn immediately after info object 1.

Now change the scale factor for each info object to 1 and re-run the program: each object takes eaxctly half of the row dedicated to these two objects - the texts are left justified. It may seem as if info object 2 does not start at the middle of the window but it does - don't be misled by the title of the window: the title IS NOT in the middle of the window. Test it with a ruler if you don't believe me.

Now change the scale factor for info object 2 to 2. It now appears at about one third of the width of the screen - info object 2 takes twice more space than info object 1, which, having a scale facor of twice the size of info object 1, is quite normal. You can try with other (integer) values...

OK, this concludes this instalment of the series. Unfortunately, due to a foreseeable increase in work, I probably won't be able to continue this as a regular series any more, i.e. one installment in each issue. I will try, however, to write more instalments, but at rather irregular intervals.

QL Forever
Wolfgang Lenerz

We're looking forward to this, Wolfgang! - Editors & Readers

Transfer from QD to Windows

Dietrich Buder

1. Foreword

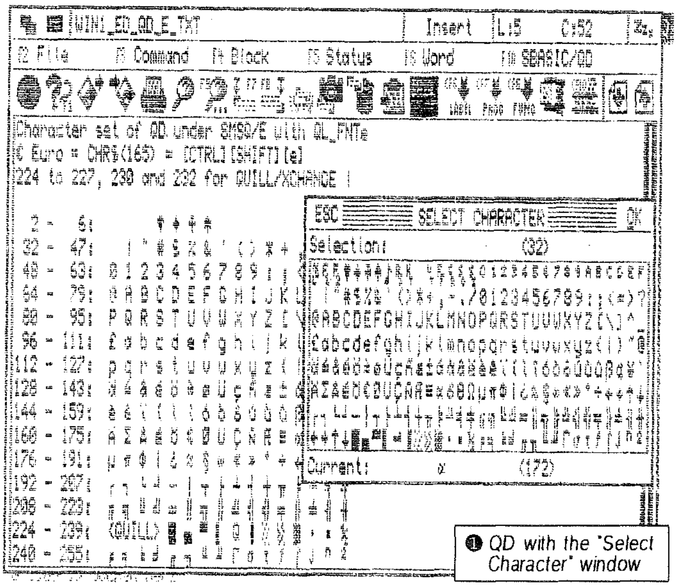
Jochen Merz wrote an article in the German QL Today Nov/Dec 1999 aboutthe transfer of QD text to Windows and asked about the necessity of such programs. My answer is 'YES AND NO'. IF a QLer writes an article for a newspaper publisher he has to give his work on a disk in DOS format for a Windows word processor, no discussion.

2. Cases

A QLer with an original QL needs this transformation, so my answer is 'YES'.

A QLer with the emulator QPC or the QXL card has a PC and it will be possible to work with a word processor of his PC, in this case 'NO'.

A QLer has a PC but needs special characters of several languages and for mathematics. He may prefer the editor QD of his QL. Why? He gets the whole character set of QD from CHR$(0)to CHR$(255) by typing [CTRL]+[t] and has no problem with going through endless menus. Typing such characters is much easier with one mouse click of QD than a look into the QL manual for the unknown keys of QUILL/XCHANGE or text87, see picture 1. The greek characters and some others are not at the normal character set of Windows WordPad and they must awkwardly be found. The Windows98 needs the way from 'Start', 'Programs', 'Utilities', 'System tools', 'Character table', 'Typeface' and 'Symbol' to find a character eg. Pi. I don't like this hard work to write one character with WordPad, and other Windows word processors may be easier. So my answer is 'Yes' too.



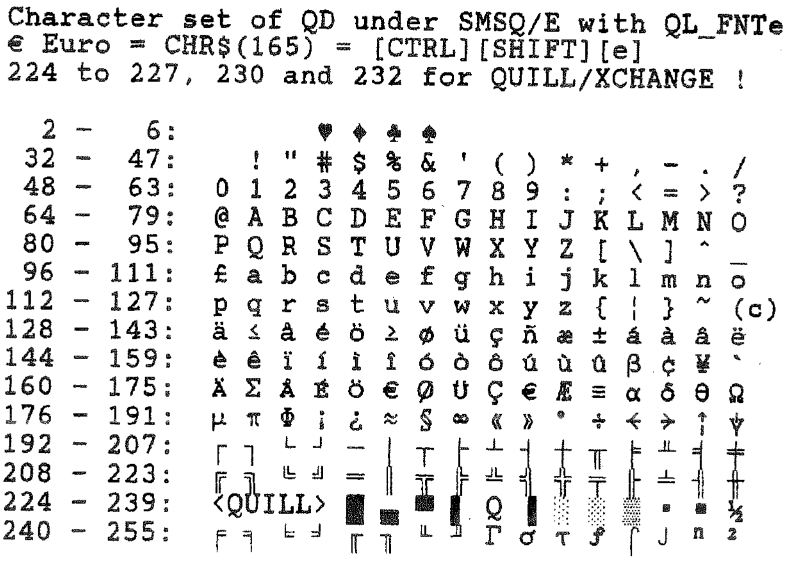
Character set of QD under SMSQ/E with QL_FNTe
€ Euro = CHR$(165) = [CTRL][SHIFT][e]
224 to 227, 230 and 232 for QUILL/XCHANGE !

❶ *QD with the 'Select Character' window*

3. More characters

If a QLer needs special characters which are not in the QL standard font then he can build his own character set with 70 new characters for QD, see picture 2. QD takes the character set from a file like 'QL_FNT', which must be configured with device and name. I take RAM3 for easy overwriting with different QL fonts. QD gives each self-made character only to the screen but a print out may be impossible.

The task is to build a special font with a font editor like QLudge from Simon N. Goodwin for the QL screen and a suitable TRA with a file like my 'TRA_EURO_E_BAS' for the printer in Standard Modus. Picture 2 (next page) shows my QD character set for normal QL work.



```
Character set of QD under SMSQ/E with QL_FNTe
€ Euro = CHR$(165) = [CTRL][SHIFT][e]
224 to 227, 230 and 232 for QUILL/XCHANGE !

  2 -   6:           ♥ ♦ ♣ ♠
 32 -  47:     !  "  #  $  %  &  '  (  )  *  +  ,  -  .  /
 48 -  63:  0  1  2  3  4  5  6  7  8  9  :  ;  <  =  >  ?
 64 -  79:  @  A  B  C  D  E  F  G  H  I  J  K  L  M  N  O
 80 -  95:  P  Q  R  S  T  U  V  W  X  Y  Z  [  \  ]  ^  _
 96 - 111:  £  a  b  c  d  e  f  g  h  i  j  k  l  m  n  ō
112 - 127:  p  q  r  s  t  u  v  w  x  y  z  {  ¦  }  ~  (c)
128 - 143:  ä  ≤  å  é  ö  ≥  ø  ü  ç  ñ  æ  ±  á  à  â  ë
144 - 159:  è  ê  ï  í  ì  î  ó  ò  ô  ú  ù  û  β  ¢  ¥  `
160 - 175:  Ä  Σ  Å  É  Ö  €  Ø  Ü  Ç  €  Æ  ≡  α  δ  θ  Ω
176 - 191:  µ  π  Φ  ¡  ¿  ≈  §  ∞  «  »  °  ÷  ←  →  ↑  ↓
192 - 207:  ┌  ┐  └  ┘  ─  │  ┬  ├  ┴  ┤  ┼  ╥  ╞  ╨  ╡  ╪
208 - 223:  ╔  ╗  ╚  ╝  ═  ║  ╦  ╠  ╩  ╣  ╬  ╤  ╧  ╢  ╫  ╬
224 - 239: <QUILL>      █  ▄  ▀  ▌  Q  ▐  ░  ▒  ▓  ▪  ■  ½
240 - 255:  ╒  ╕  ╘  ╛  ╓  ╖  ╙  ╜  Γ  σ  τ  ƒ  ⌠  J  ⁿ  ²
```

❷ QD-character set CHR$(2) to CHR$(255) by Dietrich

4. Files for transformation

The article in 'QL Today deutsch' in 1999 shows two files made in SBASIC with an error and has only a transformation of German special characters. Jochen wrote about the difference between ASCII codes of MSDOS and Windows and asked for a full translation list to help the QLers of different countries. We found better solutions for the two files and I completed the missing full translation list for my 'QL_FNT'.

I write my own programs in QDOS SuperBASIC and compile all my important files. QLiberator is the best compiler for these two transformation files. SBASIC is possible too with some changing of my source files.

5. The transformation

The transformation from QL characters to PC is very easy. Install the QLiberator file 'QLIB_RUN' (I do this in my BOOT), copy all QD text files to RAM2 (RAM1 and RAM2 are my favourites for temporary files), put a DOS format disk into FLP1 and start the file QDtoPC_1_BAS with LRUN or QDtoPC_1_OBJ with QPAC2, direct mode or with a start menu, finished. In the last case the command must be with EW, eg.

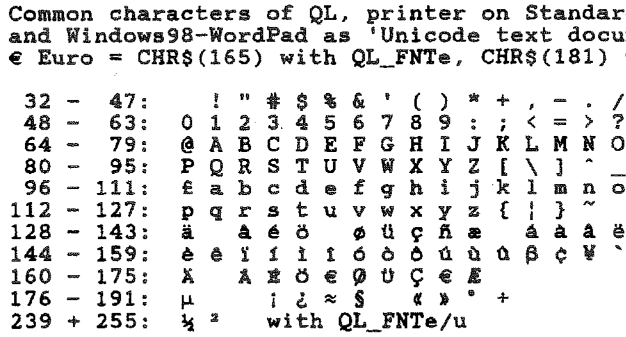
EW 'dev1_ED_QDtoPC_1_OBJ'

An error message appears if file names are too long and the program stops to let you change all wrong names.

5. Remark

The transformation needs SMSQ/E because a QL with QDOS can't read or write a DOS format disk.

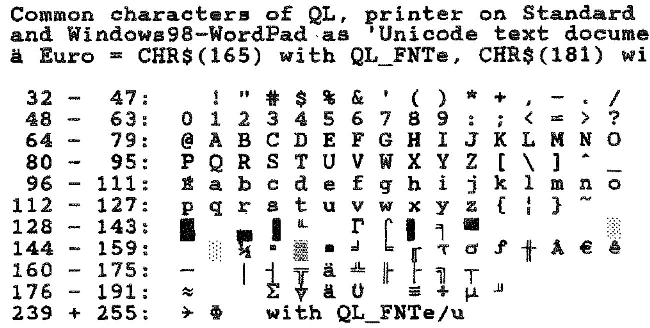
[Dietrich, it is nice that you mention SMSQ/E, but one can also use tools like the very old ATARI-DOS or XOVER to read and write DOS disks on a QL (at least DD disks) - but I fully agree that it is much easier and more comfortable with SMSQ/E! - Editor]



```
Common characters of QL, printer on Standard Mode
and Windows98-WordPad as 'Unicode text document'.
€ Euro = CHR$(165) with QL_FNTe, CHR$(181) with QL_FNTu

 32 -  47:     !  "  #  $  %  &  '  (  )  *  +  ,  -  .  /
 48 -  63:  0  1  2  3  4  5  6  7  8  9  :  ;  <  =  >  ?
 64 -  79:  @  A  B  C  D  E  F  G  H  I  J  K  L  M  N  O
 80 -  95:  P  Q  R  S  T  U  V  W  X  Y  Z  [  \  ]  ^  _
 96 - 111:  £  a  b  c  d  e  f  g  h  i  j  k  l  m  n  ō
112 - 127:  p  q  r  s  t  u  v  w  x  y  z  {  ¦  }  ~
128 - 143:  ä     å  é  ö     ø  ü  ç  ñ  æ     á  à  â  ë
144 - 159:  è  ê  ï  í  ì  î  ó  ò  ô  ú  ù  û  β  ¢  ¥  `
160 - 175:  Ä     Å  É  Ö  €  Ø  Ü  Ç  €  Æ
176 - 191:  µ        ¡  ¿  ≈  §     «  »  °  +
239 + 255:  ½  ²     with QL_FNTe/u
```

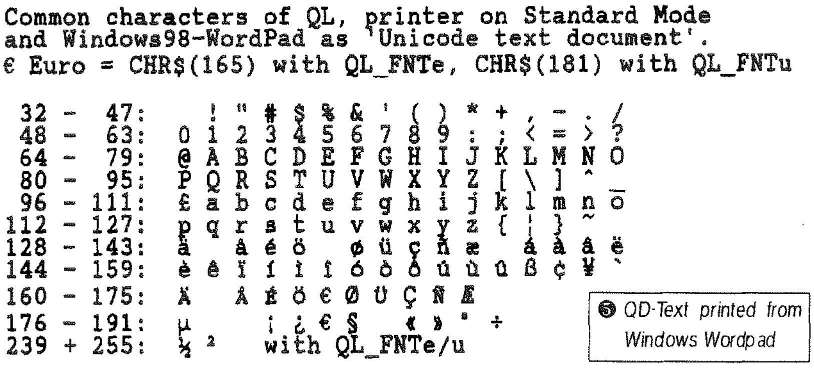
❸ QD-Text original printout from QD



```
Common characters of QL, printer on Standard Mode
and Windows98-WordPad as 'Unicode text document'.
ä Euro = CHR$(165) with QL_FNTe, CHR$(181) with QL_FNTu

 32 -  47:     !  "  #  $  %  &  '  (  )  *  +  ,  -  .  /
 48 -  63:  0  1  2  3  4  5  6  7  8  9  :  ;  <  =  >  ?
 64 -  79:  @  A  B  C  D  E  F  G  H  I  J  K  L  M  N  O
 80 -  95:  P  Q  R  S  T  U  V  W  X  Y  Z  [  \  ]  ^  _
 96 - 111:  ¤  a  b  c  d  e  f  g  h  i  j  k  l  m  n  ō
112 - 127:  p  q  r  s  t  u  v  w  x  y  z  {  ¦  }  ~
128 - 143:  █     ▄  ▌  ╙     Γ  ⌠  ▌  ┐  ▀              ░
144 - 159:     ░  ½  ▪  ▒  ■  ╛  ╘  ╓  τ  σ  ƒ  ╫  Å  €  ê
160 - 175:  ─     │  ┤  ╤  ä  ╧  ╟  ├  ╗  ┬
176 - 191:  ≈        Σ  ╥  ä  Ü     ≡  ÷  µ  ╜
239 + 255:  →  Φ     with QL_FNTe/u
```

❹ QD-Text printed from QD after transformation for Windows



```
Common characters of QL, printer on Standard Mode
and Windows98-WordPad as 'Unicode text document'.
€ Euro = CHR$(165) with QL_FNTe, CHR$(181) with QL_FNTu

 32 -  47:     !  "  #  $  %  &  '  (  )  *  +  ,  -  .  /
 48 -  63:  0  1  2  3  4  5  6  7  8  9  :  ;  <  =  >  ?
 64 -  79:  @  A  B  C  D  E  F  G  H  I  J  K  L  M  N  O
 80 -  95:  P  Q  R  S  T  U  V  W  X  Y  Z  [  \  ]  ^  _
 96 - 111:  £  a  b  c  d  e  f  g  h  i  j  k  l  m  n  ō
112 - 127:  p  q  r  s  t  u  v  w  x  y  z  {  ¦  }  ~
128 - 143:  ä     å  é  ö     ø  ü  ç  ñ  æ     á  à  â  ë
144 - 159:  è  ê  ï  í  ì  î  ó  ò  ô  ú  ù  û  ß  ¢  ¥  `
160 - 175:  Ä     Å  É  Ö  €  Ø  Ü  Ç  Ñ  Æ
176 - 191:  µ        ¡  ¿  €  §     «  »  °  ÷
239 + 255:  ½  ²     with QL_FNTe/u
```

❺ QD-Text printed from Windows Wordpad

TF Services

Compswitch

A UK 4-way trailing socket designed to switch off computer peripherals automatically when the computer is switched off, or (in the case of an ATX computer) when it auto-powers down. *Compswitch* has one control socket, and three switched sockets.

Cost.. £24

*************NEW**************

superHermes

A major hardware upgrade for the QL

All Hermes features (working serl/2 at 19200, independent baud rates/de-bounced keyboard/ keyclick) IBM AT kbd I/f // HIGH SPEED RS232 at 57600// serial mouse port and 2 other RS232 inputs// 3 I/O lines // EEPROM

Cost (including manual/software)	**£90** (£92/£93)
IBM AT UK layout Keyboard	**£11** (£13/£15)
Serial mouse	£8 (£8.50/£9)
Capslock/scrollock LED	**£1** (£1.50/£1.50)
Keyboard or mouse lead	**£3** (£3.50/£3.50)
High speed serial (ser3) lead	**£4** (£4.50/£4.50)

Hermes available for £25 (£26/£27) **Working serl/2 and independent input, debounced keyboard.**

SuperHermes LITE: All Hermes features (see above) + an IBM AT keyboard interface only.
Cost (incl keyboard lead) ....£53 (£54/£55)

QL REPAIRS (UK only)

Fixed price for unmodified QLs, excl microdrives. QLs tested with Thorn-EMI rig and ROM software.

£27 incl 6 month guarantee

Minerva

The ORIGINAL system operating system upgrade

OTHER FEATURES COMMON TO ALL VERSIONS

DEBUGGED operating system/ autoboot on reset of power failure/ Multiple Basic/ faster scheduler- graphics (within 10% of lightning) - string handling/ WHEN ERROR/ 2nd screen/ TRACE/ non-English keyboard drivers/ "warm" fast reset. V1.97 with split OUTPUT baud rates (+ Hermes) & built in Multibasic.

First upgrade free. Otherwise send £3 (+£5 for manual if requd). Send disk plus SAE or two IRCs

MKI...£40 (£41/£43) MKII...£65 (£66/£67)

MINERVA RTC (MKII) + battery for 256 bytes ram. CRASHPROOF clock & I²C bus for interfacing. Can autoboot from battery backed ram. Quick start-up.

QL RomDisq

Up to 8 mbyte of flash memory for the QL

A small plug in circuit for the QL's ROM port (or Aurora) giving 2, 4 or 8 mbytes of permanent storage - it can be thought of as a portable hard disk on a card, and reads at some 2 mbytes per second.

Think of it - you could fully boot an expanded QL, including all drivers/SMSQ etc off **RomDisq** at hard disk speed with only a memory expansion needed.

2 mbytes RomDisq	**£39** (£40/£41)
4mbytes RomDisq	**£65**(£66/£67)
8 mbytes RomDisq	**£98** (£99/£100)
Aurora adaptor	**£3** (£3.50/£4)

MPLANE

A low profile powered backplane with ROM port

A three expansion backplane with ROM port included for RomDisq etc. Aurora can be fitted in notebook case and powered off single 5V rail - contact QBranch for details. Two boards (eg Aurora and Gold Card/Super Gold Card/ Goldfire fixed to base. Suitable for Aurora (ROM accessible from outside) & QL motherboard in tower case. Specify ROM facing IN towards boards, or OUT towards back of case.

Cost .... £34 (£35/£36)

I2C INTERFACES

Connects to Minerva MKII and any Philips I²C bus

Power Driver Interface **16 I/O lines with 12 of these used to control 8 current carrying outputs (source and sink capable)**

2 amp (for 8 relays, small motors) .... **£40** (£43/£44)

4 amp total (for motors etc) .... **£45** (£48/£50)

Relays (8 3a 12v 2-way mains relays (needs 2a power driver)..**£25** (£28/£29)

Parallel Interface Gives 16 input/output lines. Can be used wherever logic signals are required.........**£25** (£27/£28)

Analogue Interface **Gives eight 8 bit analogue to digital inputs (ADC) and two 8 bit digital to analogue outputs (DAC). Used for temp measurements, sound sampling (to 5 KHz), x/y plotting**................**£30** (£31/£32)

Temp probe (-40°C to +125°C)..............**£10** (£10.50/£11)

Connector for four temp probes.............**£10** (£10.50/£11)

Data sheets..**£2** (£2.50/£3)

Control software & manual (for all I/F).......**£2** (£2.50/£3)

QL SPARES

Keyboard membrane	**no longer on sale**
1377 PAL	**£3** (£3.50/£4)
Circuit diagrams	**£3** (£3.50/£4)
68008 cpu or 8049 IPC	**£8** (£8.50/£9)
8301/8302 or JM ROM or serial lead	**£10** (£10.50/£11)
Power supply (sea mail overseas)	**£12** (£17/£21)

Prices include postage and packing (Airmail where applicable). Prices are: UK (Europe /Rest of world). Payment by cheque drawn on bank with UK address,/ postal order or CASH! I can no longer accept card payments as UK only does PDQ transaction. SAE or IRC for full list and details

10 May 01

29 Longfield Road, TRING, Herts, HP23 4DG
Tel: 01442-828254 Fax/BBS: 01442-828255
tony@firshman.demon.co.uk http://www.firshman.demon.co.uk

Here are the two Listings

```
1000 REMark ** ED_QDtoPC_1_BAS **  Jochen Merz - QL Today deutsch, Nov/Dec 1999
1010 REMark ** Adapted by Dietrich Buder - 18.04.2001 - (26.03.2001)
1020 :
1030 REMark Transformation QD text from RAM2 to FLP1 for Windows only under SMSQ/E
1040 :
1050 REMark QLIB_V3.36 input: liberate 'dev1_ED_QDtoPC_1','-noline'
1060 :
1070 from$='ram2_'
1080 :
1090 x%=SCR_XLIM: y%=SCR_YLIM: v%=40: s%=x%/12-24
1100 WINDOW #0; x%-2,v%,0,y%-v%: CLS #0: REMark lines 1100/1110 for QLIB + MEMDISP
1110 WINDOW x%-2,y%-v%,0,0: PAPER 0: INK 7: CLS: PAPER #2;0
1120 toolong$='0'
1130 PRINT \TO s%;'Transformation QD text to Windows'\\
1140 :
1150 OPEN #3; from$
1160 FOR text% = 0 TO FLEN(#3)-1 STEP 64
1170  GET #3\text%+14,name$
1180  IF name$='': END FOR text%
1190  IF LEN(name$)>12
1200   PRINT \TO x%/12-24;'Error! Name to long of:  '&name$: toolong$='1'
1210  END IF
1220 END FOR text%
1230 :
1240 IF toolong$='0'
1250  FOR text% = 0 TO FLEN(#3)-1 STEP 64
1260   GET #3\text%+14,name$
1270   IF name$='': END FOR text%
1280   dest$='flp1_'&name$
1290   IF dest$(LEN(dest$)-3)='_': dest$(LEN(dest$)-3)='.'
1300   sour$=from$&name$
1310   EX 'dev1_ED_QDtoPC_2_OBJ',sour$,dest$
1320   PAUSE 16
1330  END FOR text%
1340 ELSE
1350  PRINT  \\ TO s%;'Transformation aborted !'
1360  PRINT #0;\TO s%;'Press any key !': BEEP 3000,50: PAUSE
1370 END IF
1380 :
1390 CLOSE #3: CLS: STOP
1400 :
1410 REMark $$off
1420 :
1430 DEFine PROCedure SW
1440  SAVE_O 'win1_ED_QDtoPC_1_BAS'
1450 END DEFine SW
1460 :
1470 DEFine PROCedure SA
1480  SAVE_O 'flp1_ED_QDtoPC_1_BAS'
1490 END DEFine SA
1500 :
1510 DATA 'DRUCK',1000,1520: REMark [ALT]+[X] : print out
1520 REMark $$on
```

```
1000 REMark ** ED_QDtoPC_2_BAS ** Jochen Merz - QL Today deutsch, Nov/Dec 1999
1010 REMark ** Completed by Dietrich Buder - 30.04.2001 - (26.03.2001)
1020 :
1030 REMark QLIB_V3.36 input: liberate 'dev1_ED_QDtoPC_2','-noline'
1040 :
1050 DIM tra%(255)
1060 FOR n% = 0 TO 255: tra%(n%)=n%
1070 RESTORE
1080 REPeat sh
1090  IF EOF: EXIT sh
1100  READ ql_char, dos_char
1110  tra%(ql_char)=dos_char
1120 END REPeat sh
1130 :
1140 REPeat sh
1150  IF EOF(#0): EXIT sh
1160  BGET #0; n%: BPUT #1; tra%(n%)
1170 END REPeat sh
1180 :
1190 BEEP 300,0: STOP
1200 :
1210 DATA 96,163
1220 REMark QD can't print out QL characters at the vacant places !
1230 REMark standard printer haven't 'O tilde'=CHR$(165), therefore Euro too!
1240 DATA 128,228,          130,229, 131,233
1250 DATA 132,246,          134,248, 135,252
1260 DATA 136,231, 137,241, 138,230
1270 DATA 140,225, 141,224, 142,226, 143,235
1280 DATA 144,232, 145,234, 146,239, 147,237
1290 DATA 148,236, 149,238, 150,243, 151,242
1300 DATA 152,244, 153,250, 154,249, 155,251
1310 DATA 156,223, 157,162, 158,165, 159,145
1320 DATA 160,196,          162,197, 163,201
1330 DATA 164,214, 165,128, 166,216, 167,220
1340 DATA 168,199, 169,209, 170,198: REMark no 171-175
1350 DATA 176,181,                   179,161
1360 DATA 180,191, 181,128, 182,167
1370 DATA 184,171, 185,187, 186,176, 187,247
1380 DATA 239,189, 255,178:          REMark for QL_FNT spezial
1390 :
1400 REMark QL characters print out only with WordPad
1410 DATA 129,227, 133,245, 183,164
1420 :
1430 REMark Free translation for the last QL characters
1440 DATA 139,177, 161,255, 171,172, 172,170
1450 DATA 173,208, 174,188, 175,186, 177,215
1460 DATA 178,253, 188,175, 189,172, 190,161, 191,124
1470 :
1480 REMark $$off
1490 :
1500 DEFine PROCedure SW
1510  SAVE_O 'win1_ED_QDtoPC_2_BAS'
1520 END DEFine SW
1530 :
1540 DEFine PROCedure SA
1550  SAVE_O 'flp1_ED_QDtoPC_2_BAS'
1560 END DEFine SA
1570 :
1580 DATA 'DRUCK',1000,1590: REMark [ALT]+[X] : print out
1590 REMark $$on
```

CD Mastering for use on a QL

Darren D. Branagh

A few years ago, even the thought of being able to write to a CD (either on a PC or for a QL) was beyond the common man's grasp. However, like all technical gadgets and gismos, the price has fallen rapidly in the past few years and decent CD-RW machines can be bought for not much more than £100 now.

In this article I hope to set out how to use a CD-RW machine to master CDs for QL use, using a PC. Unfortunately, no drivers exist as yet to allow CD writing on a QL although the new Q40 can access CD-RWs via Q40 Linux. I will investigate this process in the next few paragraphs, as many people now possess a PC as well as a QL, or use the PC as a QL via the various emulator programs.

The most basic, lowest common denominator standard is ISO-9660. This places some restrictions on the filing system used to ensure a degree of compatibility between platforms. Filenames are limited to DOS-style 8.3 filenames and only 4 directory levels.

Firstly, we need to set out the method used. If you wish to use the CD on say, QPC2 for example, then you can use the ISO9660/Joliet standard or formatting and copying (I wont go into the ins and outs of this as its not important, just to highlight the fact that different computers read different formats). Basically, the ISO9660 format is good for any of the PC emulators, such as QPC2, uQLx or Q-Emulator. The 'Joliet' system is a Microsoft enhancement to ISO-9660 CDs allowing longer filenames etc, but is largely incompatible with non-Windows systems, since Amigas for example may truncate long filenames to short filenames with tildes for the abbreviations, rendering the CD difficult to use.

Other computers such as the Commodore Amiga prefer the Rockridge System (again, a variation of copying method) This was pointed out with the release of the recent Q-Celt Emulator CD, which was largely unreadable on Macs and Amigas as the Rockridge format was not used - something we hope to fix in version two. The vast amount of PC software that allows CD burning provide for the Rockridge etc. Formats and you can usually choose which one you want to use on starting your particular software - the software is usually provided free when you buy your CD-RW drive.

Macs can also use a system called HFS (Hierarchical Filing System).

The clever part is that with the right CD creation software, all of these filing systems can co-exist on the same CD to allow long filenames to be readable on several platforms - the first release of the QL Emulators CD could only really be used on systems able to handle Joliet. We hope to ensure that future versions can be used by Joliet and Rockridge systems and any other formats we can sensibly come up with.

If you want information on the various CDR formats, just do a Web search on CD formats - you'll come across a large selection of internet pages and FAQs (Frequently Asked Questions) on the subject.

So, assuming you want to use your CD on QPC2 (probably the most common setup) then you need to do the following. Its always best to create the files you want on the CD on your Hard drive first, as this reduces the rate of possible errors during burning of the CD. You can create CDs on the fly as it is called, ie. Directly to the CD, but this is not advised generally. A hard drive image first always works best.

QPC as most of you will know stores its virtual QL hard drive in a single file on the PC hard disk, called a QXL.WIN file (the name comes from the fact that the QXL card by Miracle Systems used the same file format). This file will always be the same size as the QL hard disk you see when typing DIR WIN1_ when QPC is running. If you have a 100Mb QL hard disk, the QXL.WIN file (when viewed from Windows) will be 100Mb exactly also. Therefore, the best way to store info on a CD for QL use is within a QXL.WIN file - and then just read the CD as if it were a hard disk.

So, lets assume we simply want to back up our entire QL/QPC hard drive to secure our data, or just to copy some files to CD. This is easy - simply a matter of copying the QXL.WIN to the CD. This is usually done by dragging/dropping the QXL.WIN file into you burning software and starting the burner - on a 4x Speed CD-RW machine copying a 100Mb QXL.WIN will take only a few minutes, and the newer ones are faster than this.

So, having done that, how do you restore it, assuming you have had a hard drive crash or whatever? Well, there are two ways of doing this: Firstly, you can simply put the CD in the drive, and under windows copy the QXLWIN file on it to the hard drive on the PC. Then, configure QPC to find the copied QXL.WIN file as WIN1_ on your hard drive. This can be done via the startup screen dialogue box that appears when you start QPC2.

Secondly, You can access the QXL.WIN directly from the CD. Simply configure QPC from the startup box that appears, and redefine a WIN device to be the CD-ROM on your computer. For instance, if your CD-ROM drive is drive D: when in windows or DOS, then configure WIN2_ on QPC to be D:\QXL.WIN, and it will find the CD when you DIR WIN2_.

All this may seem a bit like teaching your granny to suck eggs to a lot of you. But I still get a lot of simple questions at Shows as to how the CDs are made and how to use them, especially as a lot of people are aware that a specific device driver for a CD-ROM does not exist for the QL. This little article might just help to clarify a few things, and maybe some of you will try a CD from the Q-CELT stand at the next show - I can highly recommend the new Emulator CD (Hint, Hint!!)

The Letter Box

George Gwilt (GG) sent us a letter about Norman Dubar's (ND) assembler series. We also got Normans reply in time, and he already worked the replies into George's letter. We decided to publish it this way, as it makes reading and understanding (and exchanging compliments) easier.

GG WROTE:

There are two reasons why I find Norman Dunbar's Assembler articles useful. First they throw up errors in GWASL thus causing it to be a better program after correction. Second, they teach me things I didn't know.

ND REPLIED:

As George points out, I have, in the process of testing QLTdis, discovered a couple of foibles in GWASL. When I have reported these to George, he has always come back with a fix which is a sign of the way that things are done in the QL World. The latest version (as of 1st May 2001) is GWASL 1.17 as George spotted a bug in MOVEA before I did :o). Mind you, with George's expertise in assembler, I never expected to be able to teach him anything - so I;m well chuffed that I did.

GG WROTE:

For example, in the May/June 2000 issue, Norman demonstrated the procedure PSI_CLS. I was surprised to note that the arithmetic stack pointer held in $58(A6) was left at its value after the parameters had been fetched instead of being advanced by 6 or 8.

ND REPLIED:

I'm surprised that George didn't already know this, however, when I first started writing procedures, I always tried to tidy the maths stack as well. I figured that if I did it for procedures then I wouldn't forget to do it for functions. I learned from Andrew Pennell in the dim and distant past and his book - Assembly Language Programming on the QL - is an excellent introduction to the noble art. The need to keep a tidy maths stack is, as George mentions, a must for functions but a procedure is a lazy way to program as the QDOSMSQ system does it all for you. If only this were the case for functions.

GG WROTE:

Having looked closely at the coding for PSI_CLS, though, I thought that I might suggest a few changes which could make PSI_CLS faster. These might be of interest in general so here goes.

1. At "start" the instructions

```
    jsr   (a2)
    rts
```

could be replaced by

```
    jmp   (a2)
```

which would have exactly the same effect.
This I learnt from Simon Goodwin's DIY Kit series in QL World.

ND REPLIED:

Yes this is indeed the case, and it saved a couple of bytes as well as some time. I have used the

'JMP' version in the past, but I like to make my own returns from my own code explicit, so I tend to stick with the 'JSR' and 'RTS' combination. Both work, and in the case of another 'proper' working piece of code, the time and space saved could be useful. As the routine is called only once, I suspect that saving a few clock cycles is not all that helpful.

GG WROTE:

2. To save time the 3rd instruction in the block psi_cls could be

```
   lsr.l  d7
```

instead of

```
   divu   #8,d7
```

ND REPLIED:

Yes indeed, this is very true and much quicker as well. However, the instruction would have to be 'LSR.L #3,D7' to get the desired effect :o)

Once again, I was conscious of the fact that this could be many people's first foray into writing procedures to extend S*BASIC and as there were 8 bytes per entry in the name table, I wanted to carry that point over. It is easier for beginners to see what is going on when they see a 'divide by 8' instruction rather than a logical shift left 3 places. I believe that I have mentioned in a previous article that the shifts are equivalent to a divide by 2 (per bit shifted) but I wanted to make the 'divide by 8' point clear. When I first started doing Z80 assembler on my ZX81, I took quite a while to figure out that a shift left one bit was the same as an integer divide by two - thick or what?

GG WROTE:

3. Also the cmpi instructions could be replaced by subq instructions. Thus

```
   cmpi.w  #3,d7
```

becomes

```
   subq.w  #3,d7

   cmpi.w  #4,d7
```

becomes

```
   subq.w  #1,d7
```

The same applies to d3 later on.

ND REPLIED:

Just goes to show that there are many ways to skin a cat. The good and bad thing about assembler is that many instructions achieve the same end result. Some are quicker, some are slower but the restult is the same. This much you learn from reading other people's code - everybody has a different way of doing certain things - I like to read and learn from other people's code myself, and I highly recommend it.

GG WROTE:

4. In got_4

```
   adda.l  #2,a1   (which takes 6 bytes),
```

could be

```
   addq.l  #2,a1   (which takes 2 bytes)
```

ND REPLIED:

Well spotted, as the amount being added is in the range 1 to 8, then the ADDQ form of the ADD instruction should be used.

GG WROTE:

5. In get_rest the instruction

```
   adda.l  #6,a1
```

can be deleted. After all, the following trap #3 destroys the contents of a1.

ND REPLIED:

Again, well spotted. I must have slipped back into my old habits of keeping the maths stack tidy even in a procedure. This instruction could also have been replaced by an ADDQ, but is, as George noted, totally redundant for the reason given and also because we are in a procedure and all the parameters have been fetched so we have no further use for the maths stack.

SUMMARY

I'm grateful to George for all his help in my writing of this series - which has been going on for quite a while now - both for his comments, observations and indeed pointing out gross errors in my code and for the free donation of GWASL which has probably made this series more popular - I wish I could have had something like it when I first started programming in Assembler.

If any other readers - there are some aren't there - wish to send comments or point out errors etc, or even if I have done something that you don't understand, wish to send an email to QL Today **(editor@qltoday.com)** with all the details, then I'm sure that three things will happen. Dilwyn will have some more articles to fill up the magazine, you will have a better understanding of what I am doing and I might learn something too!

RWAP SOFTWARE

QL Cash Trader v3.7 £5

A well established accounts package for the small to medium sized business, including automatic generation of profit & loss account, balance sheet, VAT returns, reports and analysis for audit trails and management decisions. Previously sold for over £100.*

QL Payroll v3.5 £5

Manage a payroll for a small to medium sized business. Handles up to 99 employees easily, producing P45s and P60s as well as the payslips on a monthly or weekly basis. Calculates tax and national insurance and is easy to update to take account of the current tax year rules.

Q-Help v1.05 £10
Q-Index v1.04 £5

Q-Help: on-screen help for SuperBASIC commands, including TK2, Turbo Toolkit, SMSQ/E and PD toolkits. Can be used to add help to your own programs - simply produce ASCII text for each help page, add an index and Q-Help automatically cross-references and displays the links.
The PD toolkits referred to are available for £2.
Q-Index: The SuperBASIC index supplied with the Reference Manual - enter a topic such as 'screen resolution' and find out the commands which relate. Launch Q-Help for further info on the chosen command.

Sidewriter v1.08 £10

Produce landscape printouts of Easel/Qspread spreadsheets and output from QL Genealogist, as well as any other standard text file. You can specify the fonts to be used on the page. Works with all EPSON compatible printers, from 9 pin dot matrix to laser printers. A most useful utility by Dilwyn Jones - you know it must be easy to use.

ProForma ESC/P2 Drivers v1.03 £8

New improved colour and monochrome printer drivers, providing up to 720dpi for all programs written for use with ProWesS, such as LineDesign and Paragraph. Works on all Epson inkjet printers which support binary mode compression (740, 850 and 900 models at least). 1440 dpi to follow.

QL Genealogist v3.26 £20
Genealogy For Windows £50

Store your family tree for posterity. Add individuals with details of their parents and children, watch all of those links build up into a formal family tree layout. Text files and pictures may also be linked to individuals as well as notes and events, making this the perfect way to preserve the history of your family.
QL version now supports FileInfo II and QMenu as well as allowing you to link both male and female trees.
Sample tree of the Royal family since 1066 included.
PC version is event driven - enter the details as they appear in documents and it generates the tree from these. QL data and GEDCOM can be transferred to the PC version.
Both programs easy to use and complete with a step by step tutorial.

Image D v1.03 £10

Produce graphical representations of 3D objects - view them as wireframe, hidden line and shaded. Perspective and magnification can be controlled and views can be saved to file for subsequent printing. Multiple objects can be defined and positioned relative to each other. Simple to use yet produces excellent results.

SBASIC/SuperBASIC Reference Manual £40
Updates £6 each, £10 for 2 (Current Version - Rel 3)

Have you ever tried to write a program, but been lost as to the means of performing a certain action? This Reference Manual provides you with a full description and examples of how to use all of the keywords found on each of the different QLs, plus SMSQ/e, Toolkit II and many different public domain toolkits. Details of any possible problems are provided, together with descriptions of how to use the device drivers and how to ensure that your programs are compatible across the range of QL platforms.
This book is ideal for all QL users and is kept up to date with regular updates.
Orders are currently being taken for the next print run of this popular tome.
(Note: Price for the book does not include post & packing).

QL Cosmos v2.04 £5

Ever wondered what the stars in the sky looked like 100 years ago? Or, maybe you want to learn the constellations and names of what you see in the sky. This is the program for you - generates pictures of the stars and planets for any given place or time and provides details on these objects. Includes Halley's Comet, the Moon and the Solar System planets.

Q-Route v1.08C £25

The latest version of this popular route finding program. Find the quickest route or the shortest route between any two places, using roads. A wide range of maps is available for this program (see elsewhere in this advert). The program is easy and quick to use. You can even add your own places and roads to the maps to include local detail.

Flashback SE v2.03 (Upgrade only) £5

The ultimate database program - extremely fast and flexible, easy to use, updated to cope with the latest versions of the QL operating system and still maintained. A report module is included to allow you to format output in any way, including mail-merge. Unfortunately only available as an upgrade from the original version (original still available from Sector Software).

Return To Eden v3.08	£10
Nemesis MKII v2.03	£8
The Prawn v2.01	£8
Horrorday v3.1	£8
West v2.00	£5
The Lost Kingdom of Zkul v2.01	£5

A wealth of QL adventures - mainly text only.
Save the Galaxy from the ambitions of the evil dictator Nemesis.
Battle against werewolves and dracula look-alikes on a Hammer Horror set in the comical Horrorday.
Take the part of a prawn with a hangover, lost in a strange land in the hilarious Prawn.
Solve a bank-robbery by fighting the bad guys and collecting the loot in real-time old West.
Battle countless dwarves in the atmospheric Lost Kingdom of Zkul.
Return to Eden is a massive adventure over 3 disks with colourful graphics - control 3 characters in their quest to find the missing Prince.
All six adventures are available together for only £25.

A range of games to keep both the young and the young at heart amused. Some are old favourites, like Golf and a pub quiz program (500+ questions). Others are fast, colourful arcade games. Flight simulator also now available. Plenty of variation and skill required - what more can you ask for? All 6 programs only £28.

Open Golf v5.20	£8
QuizMaster II v2.07	£5
Stone Raider II v2.00	£5
Hoverzone v1.2	£5
Deathstrike v1.5	£5
Flightdeck v1.05	£10

The latest maps for Q-Route. Maps of various areas of Britain have been created by cutting them out of Big Britain Map - they will use less memory and can contain more detail. Areas covered: Scotland, NE England, NW England, S&W Yorkshire, Wales & Derbyshire, London area and South England.
Latest version of Q-Route is recommended.

Britain.map v1.11	£2
BIG Britain Map (needs 2MB) v2.02	£5
Various Area Maps of Britain (ask for details)	£2 ea
Ireland Map v1.00	£5
Belgium Map v1.01	£2
Catalonia Map v1.02	£2

D-Day MKII v3.04	£10
Grey Wolf v1.8	£8
War In The East MKII v1.24 (Upgrade Only)	£5

For the gaming enthusiast - D-Day is a classic table top wargame for one or two players - you control either the Allies or the Axis forces during WWII. With the ability to define your own army set ups and a choice of 4 different scenarios, this should keep you entertained for a while.
Grey Wolf is a graphical simulation of a submarine - can you sink the enemy shipping whilst avoiding their planes and destroyers??

RWAP Software, 26 Oak Road,
Shelfield, Walsall, West Midlands
WS4 1RQ
TEL: 01922 691607

Cheques in £sterling payable to 'R.Mellor'

* Also known as Trading Accounts

Turbo Version 4.8 - 18th April 2001

George Gwilt

Turbo compiles SuperBASIC or SBASIC programs in two steps. First Parser_task analyses the program to produce intermediate code which is then transformed by Codegen_task into the final executable program.

From the outset it was envisaged that there would be an optimiser going between Parser and Codegen which would make the resulting program even faster and probably smaller too. Simon Goodwin has given me details of what might have been in this optimiser. As a result I have produced what I call an "improver" in version 4.8 of Turbo. I call this addition an improver rather than optimiser because although I think the resulting Turbo is better, I know that further enhancements are possible. This note is designed to tell users what improvements they might expect, and to reveal to those interested the methods used to obtain these results.

Nature of Improvements

1. Printing of strings

Inside Turbo printed strings which require a line feed are printed in two steps. First the string itself and then a line feed. The enhancement is to add the character for the line feed (CHR$(10)) to the end of the string and then print it. This cuts out the call to the operating system to print one byte. Programmers should be aware that the compiled version of

```
    PRINT  a$;b$
```

requires a call to the operating system for each of a$ and b$, whereas the version for

```
    PRINT  a$&b$
```

will join together the two strings before printing by only one call. Because programmers can achieve the faster speed by their own efforts and also because the former, semi-colon, version can work for very large strings when the latter, ampersand, version may not because of overflow, this joining, or concatenating, of strings does not form part of the Turbo improvements.

2. Arithmetic

The special treatment accorded to the addition of 1 to a floating point number has been extended to the addition of 2 and 4. This would be noticed when

```
    x + 1, x + 2 or  x + 4
```

appears in an expression. The operations

```
    x% = x% + k
```

and

```
    x% = x% - k
```

where k is an explicit number, not a variable have been speeded up.

3. Conditional Branches

The sequence

```
        If_true -> A        or If_false -> A
        Jump -> B              Jump -> B
A
```

is replaced by

```
        If_false -> B       or If_true -> B
A
```

Timing

1. The percentage increase in speed is given in the table below. Under Operation is given first the operation speeded up and, in parentheses, the name of the template used. The increase differs between pseudo code (BRIEF) and native code (FAST).

2. I was not able to give figures for Conditional Branches.

Operation		BRIEF	FAST
i%=i%+k	(inc/dec_int)	68	58
f=f+1/2/4	(increment)	28	19
f+f	(duplicate_float)	15	7
x%+x%	(duplicate_int)	15	5
String	(concat_lf)	9	8

Technical Matters

Codegen

It might irritate some users that Turbo is made up of two distinct parts. That has been addressed by the use of CHARGE, which allows one instruction to start the compilation. Since the addition of yet another program in the set might have increased this irritation, I decided to make it part of Codegen itself. This means that Codegen now traverses

Parser's output file three times. During this new run it looks for the improvements discussed above and incorporates them in the ouput file.

Here comes the first decision I had to make. The output file from Parser may either be in RAM1 or, if the buffer size allocated is too small, in a file on the device chosen for the ultimate compiled program. If the file was to be extended in the course of the enhancement operation this would mean that at the least the whole of the information would have to be copied from the position of the first increase in size. But if the output was in RAM1 and the increase in size took it over the allotted buffer size, then a file would have to be created and the information set in it instead. I would really rather not get involved in such shenanigans so I decided there and then that there would be no increase in size. If an improvement required such, it would not be made. As it happens, none of the enhancements required an increase.

However, as might be expected, some enhancements cause a decrease in size. At this point I reveal that inside Turbo there is an operation called "plus_int" which is the translation of a plus sign in front of a number. It actually does nothing inside Turbo. This is because +2 and 2 give exactly the same number. It is quite reasonable for Parser to translate such pluses as "plus_int". It is less reasonable for Codegen to let them past, since this takes the time and space to jump to the next instruction. Needless to say, as part of the improvements, I have caused Codegen to ignore "plus_int" altogether. It is thus with a very clear conscience that I pad out any extra space with "plus_int"s.

So it is that the output from Parser changed by the "improver" takes exactly the same space as the original.

Library

Codegen works by looking through Parser's output and picking from a Library the templates needed for the compiled program. These templates contain the code needed to perform the operations required. Thus the code for "plus_int" is, as you have guessed, absolutely nothing. In the course of making the improvements I had to add to the templates already in existence. One of these was

```
    qinc_int_var
```

This increases a variable by an amount between 1 and 8. Those following Norman Dunbar's excellent series on Assembler Programming will realise that

```
    ADDQ  #3,(a2)
```

or something like it, will be found in the template.

One feature of Turbo is the ability to speed up sections of a program by calling for FAST rather than BRIEF code. The FAST code is operated by copying the actual code for a template rather than a mere pointer to the template copied only once. Thus if a FAST x% + 1 is wanted in several places, in each part of the resulting program, the code for this will be copied in. This certainly produces a faster though larger program.

For each new template two versions have to be written, one for the BRIEF pseudo code, and one for the FAST native code. The reason for the difference is primarily the different treatment required for parameters. For example, an integer parameter for pseudo code will be placed in the the pseudo code stream. This is accessed by the particular template which has just been called by an instruction such as

```
    MOVE.W  (A5)+,D7
```

where A5 points to the stream of pseudo code words. The remainder of the template can process this parameter as needed. In the case of native code a different approach is needed since this code itself forms part of the stream pointed to by A5. In this case arrangements have to be made to incorporate the parameter in the particular manifestation of the code itself. Hence, if the required instruction is

```
    ADDQ.W  #‹parameter value›,(A2)
```

Codegen must see that the appropriate bits of the instruction contain the parameter value. Normally this is a byte or a word and sub routines already exist in Codegen to make the insertion. Instructions to Codegen to use a particular method of insertion are placed in a set of bytes immediately following the end of the template. In the case just mentioned, I had to introduce a new sub routine.

In case a reader may be wondering how the pseudo code copes with this template, I should come clean and say that it doesn't. There is no quick way that I can see to use a value inside the pseudo code to activate an ADDQ instruction.

```
    ADDQ.W  DO,(A2)
```

does not exist!

Anyway,

```
    ADD.W  D7,(A2)
```

though longer, is just as quick. So, in this case Codegen intercepts calls to templates qinc_int_var and qdec_int_var, which increase or decrease a variable, and, if it is a question of pseudo code, alters these calls to calls for the more general inc_int_var and dec_int_var which use the instructions

```
    ADD.W  D7,(A2)
```

and

```
    SUBQ.W  D7,(A2)
```

The Double Fetch
Sometimes it will happen that Parser has requested a variable to be fetched twice in succession. In this case it is faster simply to copy the item to the maths stack (where variables are fetched to). A duplicate_int template already exists and the addition of duplicate_float was, of course, easy. These templates are used in place of the second fetch.

Answer to the Assembler Homework

```
plot_8 bsr.s   calc        ; Get A1 = screen address
       andi.w  #7,d3       ; D3 = 00000000 00000GRB
       lsl.w   #6,d3       ; D3 = 0000000G RB000000
       move.w  d3,d4       ; D4 = 0000000G RB000000
       lsl.w   #7,d4       ; D4 = GRB00000 00000000
       andi.w  #$8000,d4 ; D4 = G0000000 00000000
       andi.w  #$00C0,d3 ; D3 = 00000000 RB000000
       or.w    d4,d3       ; D3 = G000000G RB000000
       move.w  #$7f3f,d2 ; AND mask = 01111111 00111111
       andi.w  #6,d1       ; (x AND 6) in d1
       ror.w   d1,d2       ; Build correct AND mask
       ror.w   d1,d3       ; Build correct OR mask (colour)
       and.w   d2,(a1)     ; AND out the changing pixel
       or.w    d3,(a1)     ; OR in the (new) colour
       moveq   #0,d0       ; No errors
       rts                 ; All done
```

These changes have been applied to Parser_task itself, with the result that it is has been reduced in size from 76092 bytes to 74576. I did have a look to see what changes were being made and was surprised that all the enhancements mentioned above apply.

More useful bits

Jochen Merz

One of our German readers asked for articles about various subjects he is interested in. I have tried to find authors for the German issues, but apart from some promises and delays over the past 12 months or so, unfortunately nothing has arrived. So I ask the whole readership.
One of the topics he is very interested in (and maybe others too) is the creation of EPROMs for the QL. This may seem outdated for some users, but there are still QLers around with EPROM expansion boards, and there are also software commands available to simulate ROM initialisation. The EPROM Tools from J-M-S date back to this time, and they should still work. If everybody has experience how to create and handle EPROMs, what to consider (in terms of usable code etc.) - please react! I do not like to write about my own software.
As most readers keep quiet about their preferences, it is useful to have at least some topics. And I can assure you that many articles, of which the author thought that it would not interest anybody because it is too specific, has found interest and created contacts and results.

We also try to compile an up-to-date list of the various filename extensions used by QODS and SMSQ/E. If you have any (common and uncommon ones) please provide us with a text file. One line for each extension in the format extension (with or without underscore), one space, and a short explanation of the type of file. We hope to be able to provide you with a fairly complete list soon. I find myself that, after so many years, some filename extensions do not mean anything to me anymore, so it should be useful.

Another repeated item on the "wish list" of our readers is, that they would like to see examples of the use of the TK2 and the new SMSQ/E commands. The examples given in the manuals are short, and usually provide "primitive" results, but it does not seem clear how to use the procedures and functions to do something productive.

In this issue you probably have found already an article which addresses this wish, the article about PRINT_USING. Dietrichs special character article is also based on a short item I wrote in earlier issues of QL Today.

I browsed through my harddisk and watched out for more useful little utilities which show the use of less common TK2 and SMSQ/E instructions. I think I have found an interesting one for this issue.
I can't remember anymore why the following routine was programmed - I think it was done at one of the more recent QL shows where a user asked me for a solution to the problem, that he would like to find out if a specific program is running or not.
This mainly to avoid error messages when he tries to RJOB it, as RJOB is a procedure and not a function and will produce an error message if the desired job does not exist (anymore).
The following little routine does it as good as possible:

```
100 DEFine FuNction job_exist(jobname$)
110 LOCal id
120 id=0
130 REPeat
140  id=NXJOB(id,0)
150  IF id<=0:RETurn id
160  IF (jobname$ INSTR JOB$(id))=1:RETurn id
170 END REPeat
180 END DEFine
190 :
200 DEFine FuNction job_exnum(jobname$)
210 LOCal id,ct
220 id=0:ct=0
230 REPeat
240  id=NXJOB(id,0)
250  IF id<0:RETurn id
260  IF id=0:RETurn ct
270  IF (jobname$ INSTR JOB$(id))=1:ct=ct+1
280 END REPeat
290 END DEFine
```

I say "as good as possible" as scanning the job tree is something which should be done in a not interruptable operation.
The routine can fail for two reasons:

- the job is removed by a different job or terminated itself while the BASIC function searches for it.
- the job is created by a different job, one or more times.

Even scanning the job-tree twice or many more times cannot prevent this as QDOS/SMSQ/E is a multiasking operating system, in which the BASIC can be interrupted at any time without being aware of this fact.

The function is called with the jobname or part of the jobname. It will return a TRUE value (not zero) if something is found which matches (I use INSTR to do the comparison) or FALSE (zero) if nothing is found which contains this string.
You can, of course, replace line 160 by other comparisons to make sure it is the job you're looking for, e.g.

```
160 IF (jobname$==JOBS$(id)):RETurn id
```

to get case-independent string comparison, otherwise just use one "=" sign.

I have not tested the routine very carefully, as I think it was created "on the fly" at a QL show. It seemed to work at this time when we tested it.
If you do have a better, more error-proof, solution, please send it to us.
Also, if you have other useful routines, please send them in as well!

BYTS OF WOOD

SAW POINTS OFFCUTS AND SNIPPETS

Some of the things that I mentioned in the last issue with regard to hardware led me to compare the way that conversations compare between QL Users who call me when I am wearing my Q Branch hat and PC Users when I am at my day job. If a QL User wants to upgrade there is a very small range of choice that he has available but historically he has only to make minor changes to the system to dramatically increase performance. The early upgrades, such as the SANDY and early Miracle interfaces all the way to the Trump card only gave extra memory and, for the better ones at least, access to disk drives. Disk drives alone increased the viability and reliability of the QL by a vast amount but it was not until the Gold Card arrived that the performance of the computer was enhanced in real number crunching terms. By changing the processor from a 68008 to a 68000 and cramming on 2MB of memory the speed and power were hiked by an appreciable degree.
The Super Gold Card, with its 68020 and 4MB did even more for that power as well as the added luxury of the parallel port. The ten year history of

the QL thus far had seen great leaps in its abilities and, at the time of the Super Gold Card's appearance, I was living in Hamburg having first thought I would start using a PC and discarded the idea when I attempted to use one of the old XT machines. By this time the PC had been through a series of changes in format and had finally settled on DOS as an operating system. Windoze was just taking its first steps in World Domination and there was no doubt that the QL was a far better prospect.

State of the Art or Art of the Salesman?

From this point on the paths that the two machines followed were widely different. The PC first brought out the 486 processors which required new motherboards. Next it changed the memory to 72 pin SIMMs and added PCI slots.

It seems to be at this point that the PC industry went into new format overdrive. The introduction of the first Pentium chips saw the emergence of a new socket for the processors - known as the Super socket Seven.

In the Super Socket Seven days you could plug any of a large range of processors into the motherboards Cyrix, AMD and the Intel all had CPUs which would fit most motherboards so upgrading was not too difficult - then. Intel made the first leap into a new format with its Pentium II - a CPU that resembled a small Instamatic Camera and needed a new (slot 1) mounting. AMD leaped in with the Athlon and a new mounting again (slot A). Intel went back to a socket mounting but this time it was a socket 370 so AMD came up with the socket A and then along came the pentium 4 ......

Confused? Well add to that the changing of the Graphics slot to AGP (types 1 through to 4) and the new CNR and AMR modem and network slots and you get to the state where you have to sit down and have a cup of tea and a digestive biscuit. Oh and I did not mention Power Supplies and Cases did I? The original AT case and PSU gave way to an ATX which was all very well until you get to the Athlon slot 1 which needed more power so a new PSU was needed and then along came socket A Athlons which needed even more so ATX v 2.03 was born and the current Pentium 4 motherboards have an extra plug on them so thats a new PSU again and the mounting of the heat sink for the chip needs to be screwed onto the case so you need a new case too, and different Memory modules, and, probably, a new graphics card because the P4 will only work with a small range these - but don't buy one of those yet because there is a new P4 coming out soon and who knows what that needs? On second thoughts forget the tea and digestives and pass me the Prozac.

Meanwhile, In the Black Corner.........

Throughout all of this the QL made little changes to its hardware and all of these, with the exception of the Aurora, could be incorporated onto the original motherboard. Hermes, Minerva, superHermes and the ROMDisq all work very happily with the old motherboard and all of the expansion devices plugged neatly into the slot at the end. Even the proposed GoldFire will do that. Admittedly the graphics were stuck in 512 x 256 mode until the Aurora can along and the maximum number of colours remained stubbornly at 8 until recently but, if you had a QL you could plug most stuff into it and it worked pretty well.

The first major upgrade to the QL hardware was the Q40 and it was that leap which, as I mentioned last issue, made colour drivers a reality. The concept of discarding all of their accumulated QL hardware to trade up to a Q40 was a bit alien to most users but all that have taken the plunge have been very happy with it.

This brings me back to the point I started off with. If a QL user calls me about an upgrade the conversation is very simple. How much? Do I have one? Will it still run the older software. When a PC user calls we start off with a conversation about an upgrade. After I have asked him what he has at the moment the conversation usually turns into buying a new machine completely and the only thing he has left that he can use from the old one is a floppy. Most people tend to give up at this stage and buy a new machine. They pass the old one down to their wife, who passes her one on to the children and we wind up in the state where the dog has two PCs in his kennel. Can't call you - Fido is online downloading pictures of poodles in compromising situations.

Most of these people complain that they 'only bought the machine a couple of years ago and it cost over £1000.00. The salesman said it was state of the art'. They want to know if the new machine they buy is futureproof but how can I tell them that when I don't know what new piece of hardware mayhem is on the horizon? Future proof only means it will not look too shabby next week as far as I can tell.

Roadmaps and Dead Ends

Intel are always talking about their 'roadmaps for the next generation of devices' and this tends to give the impression that they have some masterplan which the PC industry is embarked upon to bring a whole 'new computing experience' to the public. This seems to forget that, for the majority of the computing public, that experience is mounting frustration and a great need to force the giant computing companies off the road and indulge in a little cybernetic road rage. The rapid change in the PC hardware formats have left many programmers in a tail spin - unable to keep their products functioning correctly across such a wide range of hardware platforms and 'standards'.

Among the people who have bought new systems from the company I work for was one whose new 1GHz Athlon machine would not go up a level on a current football game because it did not like the ATi 32Mb graphics card. Another well known of CD writing software would completely trash the O/S if installed on certain systems. It is hard to blame the programmers here because they have a big job to do and deadlines for release are so tight. Back in the old days of the QL programmers used to complain that they could not test their software on JM, JS and Minerva ROMs so you can imagine what problems beset the mainstream PC boys.

This shifting hardware target is a root cause of the inherent instability of most PCs and when you add this to the acknowledged fact that the root O/S it all sits on has more bugs in it than the insect department of London Zoo you can see why the system falls apart so often.

QL Vadis

So where does that leave us in QL land? The answer is not as simple as it may seem. The lack of changes in the hardware, with the exception of the Q40, over the past few years has led to a very stable system and most programmers writing for it know exactly what they can do to get things working. The tight memory constraints has led to concise code and programs and data which can fit comfortably on an HD floppy - because we have no CD ROM.

The downside is that people do not write programs just for profit or fame - there is little of that for QL programmers these days. There is also a sense of challenge and creation and that drive is, I suspect, greater than that of the market forces. Some people have said that the colour drivers were a waste of time and effort because data comes in black and white but that misses the point. If a programmer is forced to sit in a groove and is constrained by a lack of resources he will soon start to feel frustrated and I suspect that is why some the more innovative of our programmers have migrated to LINUX.

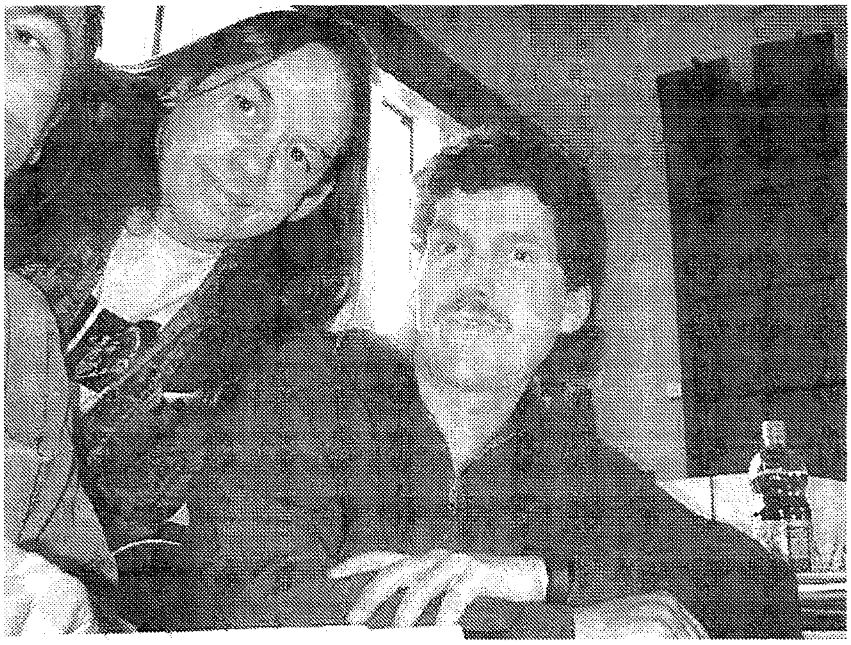
A recent contribution on the QL-Users list when discussing viewing and converting graphics files on the QL ram 'I have no use for this kind of program because I keep my QL for serious work'. That is all very well but, in my opinion, serious work is something you want to do and if you seriously want to look at and convert graphics files on your QL and to use a digital camera then that falls into the category of serious work and you have every right to try to find out how to do it - and some people have been doing just that.

Simon Goodwin Snaps

Towards the end of the Quanta AGM at Portishead Simon came over to Jochen and I and began to try out the latest version of his software for digital cameras. He had been demonstrating this in the centre of the hall for most of the afternoon but Jochen and I had both been too busy to get over to see it. He plugged his Kodak 215 Camera into Jochen's Atari TT and ran the software which refused to play ball. Jochen has often said that the TT is probably the most temperamental hardware to run things on so he was not surprised but Simon fired a series of questions about serial ports at Jochen and, after a moments reprogrammming announced that he had it all worked out.

He placed the camera on the table and I wandered over to take a look. Simon's software does the whole job so you have no need to touch the camera at all. He ran the program as I looked over Jochens shoulder and the result was a picture on the screen in a matter of moments. The picture was only running in the four colour mode but would have been pretty good for all that if the subject matter, Jochen and I, had been any better. We transferred the picture onto a floppy and used Dave Westbury's excellent Photon to display it on my Q40. The result was a pretty good full colour photo. Great stuff Simon!

I gather the software will only run on the Kodak cameras and that the one to go for is the 215 because it uses a serial



port for data transfer instead of the more usual USB. These are, however, among the cheapest of the Kodak cameras and good value for money.

The Left hand does not know what the Right hand is printing

Another revelation at the AGM was Mr. M. Smith's Epson Stylus 880 printer. You may recall that both Jochen and I discussed this printer a while ago saying that it was one of the few new printers available from EPSON which use the ESC/P2 printer codes - vital for use with a QL. A customer of mine asked me about getting one for him and I was looking into pricing when he called me back to say that he had spoken to the help desk and they say that the website was wrong to say that it used ESC/P2. I called them myself to make sure and they told me the same story. We therefore printed a quick retraction so people did not buy printers which did not work.

Mr. Smith had, however, bought one and found that it printed just fine. We had discussed this on the telephone a few times and he had sent me sample printouts and the printers control test page. Jochen and I discussed this matter on the journey to Portishead because we feel that the printer issue is one which concerns a lot of users and we were pleased to see that Mr. Smith had made the journey from his Stratford Bakery to the AGM. We were even more pleased to find that he had brought his printer so we quickly plugged it in to both the TT and the Q40 to test it.

I am happy to report to you that it prints perfectly from QL programs that have the EPSON codes available. Colour printing is possible from programs that support that function and it printed perfectly from ProWesS using the colour replacement drivers. This is a high quality unit at a very good price (2880 dpi) and it is probably the quietest EPSON printer I have ever seen. If you need a new printer I recommend this one without reservation.

It seems that EPSON do not know what their own printers do.

Just in Case

Drifting back to the hardware issues for a moment, it seems my comments in the last issue about the Q40/Q60 development caused some consternation among my readers. I had no intention to cast any sort of doubts on the viability of these two boards. I was really trying to point out the problems that small manufacturers of hardware have - especially when they are trying to do things on a shoestring with a very limited amount of customers.

My points were that the change in the power supply from AT to ATX meant that some tinkering with actual power socket of the PSU needed to be done to get the case to switch on and off. Thierry Godefroy has solved this by changing the ATX PSU for an AT one but this works only in 80% of the cases. Some of the PSUs have slightly different mountings or sizes. Beware of the PC 'Standard' (a standard is also something which flaps around in the breeze). I was also unsure as to how well a board would fit into an ATX case - would the mounting holes line up OK? It seems that they do and this has been confirmed by Thierry and Peter Graf so that is OK. Someone also mentioned cases which have dual AT and ATX backplates so these would be a good thing to look out for if you want to embark upon a construction of this kind.

My final point was about availability of older parts. True there are AT cases available but many of these are pretty awful. Most of the good manufacturers now make only ATX cases. Bad cases often have even worse PSUs - I bought one which caught fire straight from the box! One of my friends had his entire office server destroyed by a faulty PSU so do spend a bit of time and effort in getting something good. As a rule of thumb the cheaper the case you buy the bigger the hole you dig for yourself later.

Re-Flex Actions

There is a new format for PC cases called 'Flex ATX' and this may be a very good solution for people who want to get a Q40 into a small case. The 'Flex' format is designed to take the micro ATX motherboards which have sound, graphics and other facilities on board. There are quite a few of these around now that you can get a whole soundcard on a single chip. The drawback of this format for PC users is that it will only use the half-height plug in cards and there are very few of these around so adding extras is difficult. For the Q40/Q60 the format is almost perfect given the fact that you can rig the ATX PSU. The I/O card that we use on the Q40 is a half height one so you only need to change the backplate for the one provided by the case manufacturer. You will, of course still have to modify the keyboard socket but the case is so small that the result will be a very neat unit. It will only take a single floppy, hard disk and a CD ROM so Q40/Q60 users will have to decide between having a CD drive for LINUX use or the two floppies they are more used to but, apart from that it is a very good housing. The one I have been looking at is the AOpen 'Flex ATX' case made by Acer.

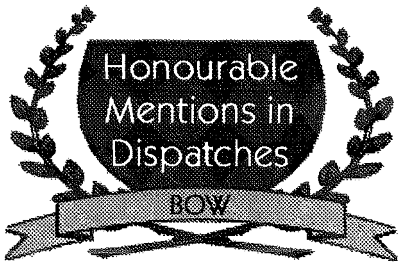
There's a Hole in My O/S, Dear Liza......

When we met to go to the AGM last month Jochen told me that TT had discovered a bug in the Open Console routines of all versions of SMSQ/E post colour drivers and that this was the most likely cause of the problems that the O/S has been having with QLiberated programs. Not being one of those blessed with the knowledge of programming that TT and Jochen have I was a bit in the dark but it seems that the problem is such that it will give different symptoms depending on what is loaded at the time and a whole host of other variables.

TT has been trying to fix the bug but beta versions sent to Jochen have caused more problems than they have solved so he is still hard at work trying to pin it down. Personally, I am glad that the fact that there is a bug has been recognised because it means that a solution will, eventually, appear. It drove me crazy for a long time.

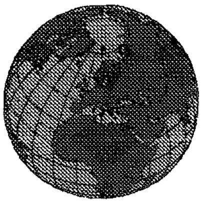
Then Patch it Dear Johnny.......

I was going to award the honourable mention to Richard Zidlicky this issue because I read that he had released a patch for the Q40 version of SMSQ/E which allowed troublesome QLib programs to run properly. I downloaded the patch from Thierry's site and tried it out. The result, on my system at least, was either a hang up or a nasty crash which reset the system clock to some time in the future and cause many other odd effects. I suspect that these effects are due to interaction between the patch and the other programs I have loaded at the time but I am interested in hearing from other people who have managed to get the patch installed and have run it successfully with their systems. I would very much like to find out why it does not work with my particular setup too.

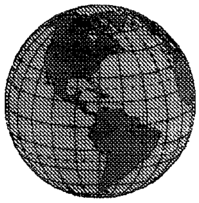


Honourable Mentions in Dyes Patches

Richard was pipped at the post this issue by Wolfgang Lenerz for his tiny, but perfectly formed, col_obj. This program seems to have no discernible name but can be found, labelled 'Col_zip' on Thierry's website. It is a small Qliberated program which displays a square with three scroll bars around the right, left and bottom frames. Move the sliders and the square changes colour. The best bit is the three sets of numbers at the very bottom. These change as you move the slider and, if you enter these into a ProWesS configuration parameter you get the colour show in the square. Simple but wonderful too. My ProWesS has been through many violent shades of Lime Green, Puce Orange and Pavement Pizza. I downloaded it late at night and intended to just have a quick look before going to bed - two hours later I was still fiddling with it !



The QL Show Agenda



QL Meeting - (NL) Eindhoven

Saturday, 23rd of June
Held at its usual Venue: St. Joris College.
The meeting starts at 10am and ends at 4pm.
J-M-S will be there, QBranch and TF Services definitely promised to be there (2 cars available)!

Quanta workshop - (GB) Byfleet

Sunday, 23rd of September
Same venue as every year:
The Byfleet Village Hall.
More details will follow in the next issue

QL Meeting - (D) Berchtesgaden

Sat./Sun., 6th/7th of October
Details inside this issue!
Definitely a very nice, social event too!
South German/Austrian meetings are always more than "just" a QL meeting.

QL Show - (F) Paris

Saturday, 13th of October
Same venue as last year - Paris University.
Details will follow in the next issue.